Gateway Customization Guide

Red Hat Directory Server

Version 7.1

April 2005

Red Hat, Inc.

1801 Varsity Drive
Raleigh NC 27606-2072 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park NC 27709 USA

© 2001 Sun Microsystems, Inc. Used by permission. © 2005 by Red Hat, Inc. All rights reserved. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at http://www.opencontent.org/openpub/).

Distribution of substantively modified versions of this document is prohibited without the explicit permission of the copyright holder.

Distribution of the work or derivative of the work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from the copyright holder.

Red Hat and the Red Hat "Shadow Man" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.

All other trademarks referenced herein are the property of their respective owners.

The GPG fingerprint of the security@redhat.com key is:

CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

Contents

List of Tables **13**

List of Figures **15**

Preface **17**
Prerequisite Reading 17
Conventions Used in This Book 18
Related Information 18

Chapter 1 Introduction **21**
What Is a Gateway? 21
Gateways Installed with Directory Server 22
Default Gateway 22
Directory Express 23
HTTP Server Requirements for Gateways 23
Directory Express and Default Gateway 23
Directory Express (pb.conf) 23
Default Gateway (dsgw.conf) 25
Support for Multiple Gateway Instances 25
Specifying Gateway Configuration to Gateway CGIs 26
GET Operations (GCONTEXT) 26
POST Operations (PCONTEXT) 26
URLs for Gateway Access 27
Anonymous and Non-Anonymous Searching 27
binddnfile 27
Automatic Updates to Directory Configuration 27

Chapter 2 Setting Up the Gateway 29
Gateway Installation Planning 29
Location of Gateway Files 29
Gateway Cloning 30
Securing Gateway Configuration and Settings 30
Protecting Bind DN and Password 31
Protecting Root Processes on UNIX Systems 31
Updating the Gateway with Changes to Directory Server Configuration 31
HTTP Server Recommendations for Directory Server Gateway 31
Running the Gateway in High-Usage Networks 32
HTTP Server Configuration 32
Name Translation Mapping 33
Gateway Root Suffix 33
Configuring the Gateway for Web Servers 33
Creating a New Gateway Instance 36
Gateway Cloning 37
Gateway .conf File Configuration 37
Changing the Default Port Setting 37
Setting Up a Directory Manager for the Gateway 38
Configuring the Directory Manager DN 38
Authenticating as Directory Manager 39
Setting Up the Suffix for Adding Entries 39
Setting Up SSL Support 40
Configuring the Gateway to Use SSL 40
Setting vCard Properties 40
Configuring Gateway Clients 40
Language Support for HTTP Clients 41
Unicode and Latin-1 Character Sets 41
Displaying a Non-English Alphabet 41
Configuring Netscape 7.x for Preferred Language 41
Customizing Communicator's LDAP Settings 42

Chapter 3 Gateway Localization 43
Unicode and Support for UTF-8 43
How the Gateway Selects a Character Set 44
How the Gateway Selects from Multiple Requested Characters Sets 44
HTTP Clients that Request UTF-8 44
HTTP Clients that Do Not Request UTF-8 45
Special Characters 45
Non-Breaking Space 45
Query Strings 45
Gateway Locales 46
Support for Multiple Locales 46

Setting Up Locales for Translation 46
dsgw-l10n.conf 47

Chapter 4 File Controlling Gateway Functionality 49
Files Controlling Gateway Functionality 49
Gateway .conf Files 51
dsgw.conf 51
pb.conf 51
Gateway Search Configuration Files 51
Object Class Templates 52
Gateway Script Files 53
Gateway Search Result Templates 53
Banner Files 54

Chapter 5 Editing Entry Types 55
Entry Types (Object Classes) 55
Parameters Controlling Entry Types 55
template 56
location 56
newtype 56
Considerations for Adding New Entry Types 57
Mapping Locations and Entry Types 57
Mapping Entry Types to Locations 57
Setting Up Organizational Units 58
Configuring DN Formats for Entry Types 59
UID-Based DN 59
Modifying the Default DN Format 59
Object Class Attributes in Template Files 59
Default Gateway Object Classes 60
Templates and Directives 60
Entry-Related Directives 60
Adding Attributes to Object Classes 61
Deleting Attributes from Object Classes 61
Extending Object Classes 62
Adding a Template for a Child of a Parent Class 63
Creating a New Parent Object Class 64

Chapter 6 Search Attributes, Filters, and Results 67
Search Configuration Files 67
dsgwsearchprefs.conf 67
dsgwfilter.conf 68
Changing Search Scope 68

Modifying Search Attributes for Advanced Searches 69
Standard and Advanced Searches 69
Specifying Search Attributes for Person 72
Directory Express Search Support for User ID 73
Adding Search Support for Additional Attributes 73
Adding Search Support for a New Object 74
Extending Search Preferences 74
Creating a New Search Object 75
Modifying Default Search Filters 76
Search Filters for User Data Patterns 77
Specifying a Search Filter for a New Object 77
Customizing Search Result Templates 78
How the Gateway Displays Search Results 78
Search Result Tables 78
Modifying Search Result Templates 80
Adding Information to Search Results 80
Removing Information from Search Results 81

Chapter 7 Customizing Graphics and Color 83
Appearance of Gateway Forms 83
Banner Image 83
Updating the Banner Image (title.gif) 84
Changing Dimensions of Banner Image 84
Changing Filename of Banner Image 84
Button Images 84
Updating Button Images 85
Color Schemes 86
Files Controlling Colors on Gateway Forms 86
Changing Colors Using BODY Tag 87
Changing Colors Using Directives 88
Changing Table Colors 88
Specifying Color Names and Color Codes 88
Changing Color of Table Headings 89
Changing the Background Color of Table Rows 90

Appendix A Parameters Defined in the .conf File 91
authlifetime 91
Format 91
Example 91
baseurl 92
Format 92
Example 92
binddnfile 93

Format . . . 93
Example . . . 93
changeHTML . . . 93
Format . . . 93
Example . . . 93
charset . . . 93
Format . . . 94
Example . . . 94
configdir . . . 94
Format . . . 94
Example . . . 94
dirmgr . . . 95
Format . . . 95
Example . . . 95
enable-aim-presence . . . 95
Format . . . 96
Example . . . 96
gwnametrans . . . 97
Format . . . 97
Example . . . 97
htmldir . . . 97
Format . . . 97
Example . . . 97
ignoreAcceptCharsetFrom . . . 97
Format . . . 97
Example . . . 98
include . . . 98
Format . . . 98
Example . . . 98
location . . . 98
Format . . . 98
Example . . . 99
location-suffix . . . 99
Format . . . 99
Example . . . 99
newtype . . . 100
Format . . . 100
Example . . . 101
NLS . . . 101
Format . . . 101
Example . . . 101
orgchart-attrib-farleft-rdn . . . 101
Format . . . 101

Example . . . 101
securitypath . . . 102
Format . . . 102
Example . . . 102
template . . . 102
Format . . . 102
Example . . . 102
url-orgchart-base . . . 102
Format . . . 103
Example . . . 103
vcard-property . . . 103
Format . . . 103
Example . . . 105

Appendix B Gateway Directives . . . 107
Introduction . . . 107
Structure of an HTML Template . . . 108
Structure of an HTML Template for Directory List . . . 108
Context-Related Directives . . . 109
GCONTEXT . . . 109
Arguments . . . 109
Example . . . 109
PCONTEXT . . . 109
Arguments . . . 109
Example . . . 109
Entry-Related Directives . . . 110
DS_ENTRYBEGIN . . . 110
Arguments . . . 110
DS_ENTRYEND . . . 110
Arguments . . . 110
DS_ATTRIBUTE . . . 110
Arguments . . . 110
Examples . . . 113
DS_OBJECTCLASS . . . 114
Arguments . . . 114
Example . . . 114
DS_VIEW_SWITCHER . . . 114
Arguments . . . 114
Example . . . 115
DS_SORTENTRIES . . . 115
Arguments . . . 115
Example . . . 115
DS_SEARCHDESC . . . 115

Arguments . . . 115
DS_POSTEDVALUE . . . 115
Arguments . . . 115
Example . . . 116
DS_EDITBUTTON . . . 116
Arguments . . . 116
Example . . . 116
DS_DELETEBUTTON . . . 116
Arguments . . . 116
Example . . . 116
DS_SAVEBUTTON . . . 117
Arguments . . . 117
Examples . . . 117
DS_EDITASBUTTON . . . 117
Arguments . . . 117
Example . . . 117
DS_NEWPASSWORD . . . 117
Arguments . . . 118
DS_CONFIRM_NEWPASSWORD . . . 118
Arguments . . . 118
DS_OLDPASSWORD . . . 118
Arguments . . . 118
DS_HELPBUTTON . . . 118
Arguments . . . 118
Example . . . 118
DS_CLOSEBUTTON . . . 118
Arguments . . . 119
Example . . . 119
DS_BEGIN_ENTRYFORM . . . 119
Arguments . . . 119
DS_END_ENTRYFORM . . . 119
Arguments . . . 119
DS_EMIT_BASE_HREF . . . 119
Arguments . . . 119
DS_DNEDITBUTTON . . . 119
Arguments . . . 120
DS_BEGIN_DNSEARCHFORM . . . 120
Arguments . . . 120
DS_ATTRVAL_SET . . . 120
Arguments . . . 120
Example . . . 120
IF/ ELSE/ ELIF/ ENDIF . . . 120
Arguments for IF and ELIF . . . 120

Arguments for ELSE and ENDIF 121
Examples 122
Miscellaneous Directives 122
BODY 122
Arguments 122
Examples 122
COLORS 122
Arguments 122
Example 122
TITLE 123
Arguments 123
Example 123
ENDHTML 123
Arguments 123
HELPBUTTON 123
Arguments 123
Example 123
INCLUDE 123
Arguments 123
Example 124
INCLUDECONFIG 124
Arguments 124
Example 124
DS_LAST_OP_INFO 124
Arguments 124
Example 124
DS_LOCATIONPOPUP 124
Arguments 125
Example 125
DS_GATEWAY_VERSION 125
Arguments 125
Example 125
IF/ ELSE/ ELIF/ ENDIF 125

Appendix C CGI Usage 127
auth 127
lang 127
search 127
csearch 127
unauth 128
dnedit 128
edit 128
doauth 128

domodify 128
newentry 129
dosearch 129

Index 131

Preface

The descriptions, instructions, and examples in this guide can be used to create and modify a gateway instance to suit the needs of users in your organization. The preface contains the following sections:

- Prerequisite Reading (page 17)
- Conventions Used in This Book (page 18)
- Related Information (page 18)

Prerequisite Reading

This guide explains Red Hat Directory Server Gateway and is intended for anyone who needs to implement a simple gateway instance with basic directory lookup functionality. It is also for users who wish to implement a more powerful gateway instance with directory authentication and administration capability.

This guide assumes the reader is familiar with HTML file syntax and has a rudimentary understanding of how the LDAP directory stores information. The guide does not describe many of the basic directory and architectural concepts that you need to deploy, install, and administer your directory service successfully. Those concepts are contained in the *Red Hat Directory Server Deployment Guide* and *Red Hat Directory Server Administrator's Guide*. You should read those books before continuing with this manual.

When you are familiar with Directory Server concepts and have done some preliminary planning for your directory service, you can install the Directory Server. The instructions for installing the various Directory Server components are contained in the *Red Hat Directory Server Installation Guide*.

Also, *Managing Servers with Red Hat Console* contains general background information on how to use the Red Hat Console. You should read and understand the concepts in that book before you attempt to administer Directory Server.

Conventions Used in This Book

This section explains the conventions used in this book.

- `Monospaced font` — This typeface is used for any text that appears on the computer screen or text that you should type. It is also used for filenames, functions, and examples.
- Notes and Cautions text boxes.

NOTE Notes and Cautions mark important information. Make sure you read the information before continuing with a task.

- The greater than symbol (>) is used as a separator for successive menu selections. For example, Object > New > User means that you should pull down the Object menu, drag the mouse down to highlight New, and drag the mouse across to the New submenu in which you must select User.
- Throughout this book you will see path references of the form:

  serverRoot`/slapd-`*serverID*`/...`

  serverRoot is the installation directory. The default installation directory for UNIX is `/usr/redhat/servers`. On Windows, it is `c:\redhat\servers`. If you have installed Directory Server in a different location, you should adapt the path accordingly.

  serverID is the ID or identifier you assigned to an instance of Directory Server when you installed it. For example, if you gave the server an identifier of `phonebook`, then the actual path would look like this:
  `/usr/redhat/servers/slapd-phonebook/. . .`
- All paths specified in this manual are in UNIX format. If you are using a Windows-based Directory Server, you should assume the equivalent file paths whenever UNIX file paths are shown in this book.
- In examples/sample code, paths assume that the Directory Server is installed in the default location `/usr/redhat/servers`. If you have installed your Directory Server in a different location, adapt the paths accordingly.

Related Information

The document set for Directory Server also contains the following guides:

- *Red Hat Directory Server Deployment Guide*. Provides an overview for planning your deployment of the Directory Server. Includes deployment examples.
- *Red Hat Directory Server Installation Guide*. Contains procedures for installing your Directory Server as well as procedures for migrating your Directory Server.
- *Red Hat Directory Server Administrator's Guide*. Contains procedures for the day-to-day maintenance of your directory service. Includes information on configuring server-side plug-ins.
- *Red Hat Directory Server Configuration, Command, and File Reference*. Contains information about using the command-line scripts shipped with Directory Server.
- *Red Hat Directory Server Schema Reference*. Contains information about the Directory Server schema.
- *Red Hat Directory Server Plug-in Programmer's Guide*. Describes how to write server plug-ins in order to customize and extend the capabilities of Directory Server.
- *Red Hat Directory Server Org Chart.* Introduces the Red Hat Directory Server Org Chart application and explains how to integrate it with an instance of Directory Server.
- *Red Hat Directory Server DSML Gateway Guide.* Introduces the Red Hat Directory Server DSML Gateway function and explains how to customize it for use as an independent gateway.

For a list of documentation installed with Directory Server, open the *serverRoot*`/manual/en/slapd/index.htm` file.

For the latest information about Directory Server, including current release notes, complete product documentation, technical notes, and deployment information, check this site: `http://www.redhat.com/docs/manuals/dir-server/`

Chapter 1

Introduction

This chapter describes the gateway functionality of Red Hat Directory Server (Directory Server). The chapter contains the following sections:

- What Is a Gateway? (page 21)
- Directory Express and Default Gateway (page 23)
- Support for Multiple Gateway Instances (page 25)
- Anonymous and Non-Anonymous Searching (page 27)
- Automatic Updates to Directory Configuration (page 27)

What Is a Gateway?

A gateway is an HTTP-to-LDAP client that lives on an HTTP server. Using special directives embedded in HTML files, a gateway allows users to access user directory data using any kind of web browser. Using a gateway does not require login to the Red Hat Console.

In Directory Server, many gateway instances can be defined on one HTTP server, providing access to any number of Directory Servers. A gateway instance consists of the following:

- A `.conf` file, stored in the *serverRoot*`/clients/dsgw/context` directory, defining the context for gateway instance. For example, `dsgw.conf` defines the *dsgw* gateway instance.
- An HTML directory for object class templates and other files containing gateway directives used to communicate with Directory Server.
- A configuration directory for directory search, directory authentication, language files, and gateway scripts.

Gateways Installed with Directory Server

Two gateway instances are installed during Directory Server installation: the default gateway and Directory Express. Both gateways are configured to use the suffix set when the Directory Server was configured and non-SSL (Secure Socket Layer) communications.

NOTE A Java gateway based on the DSML protocol is also installed with Directory Server. For more information on using and configuring the DSML gateway, refer to the *Red Hat Directory Server DSML Gateway Guide*.

Netscape
Server
Products
Services for Users
Directory Server Express
Search for users by name, user ID or extension.
Directory Server Org Charts
Browse org charts of your organization.
Services for Administrators
Directory Server Gateway
Search for and edit directory entries.
Netscape Server Central
Check for upgrades and information about Netscape server products.
Netscape Administration Express
View server status and configuration/log data.

For more information, see “Location of Gateway Files,” on page 29.

Default Gateway

The configuration file for the default gateway is at *serverRoot*`/clients/dsgw/context/dsgw.conf`.

Following Directory Server installation, the default gateway can be accessed from `http://`*adminHost*`:`*adminPort*`/`

or by directly using this URL:

`http://`*adminHost*`:`*adminPort*`/clients/dsgw/bin/lang?context=dsgw`

Directory Express

The configuration file for Directory Express is *serverRoot*`/clients/dsgw/context/pb.conf`.

During Directory Server installation, Directory Express is configured to use the Red Hat Administration Server installed with the directory as its HTTP server.

Following Directory Server installation, Directory Express can be accessed from `http://`*adminHost*`:`*adminPort*`/`

or directly using this URL:

`http://`*adminHost*`:`*adminPort*`/clients/dsgw/bin/lang?context=pb`

HTTP Server Requirements for Gateways

A gateway instance requires an HTTP server that can communicate with Directory Server. For optimum performance and highest security, the gateway should be configured to run under a high-performance HTTP server, such as the Red Hat Enterprise Server.

For more information, see “HTTP Server Configuration,” on page 32.

Directory Express and Default Gateway

The following sections describe Directory Express and the Default Gateway in detail:

- Directory Express (pb.conf)
- Default Gateway (dsgw.conf)

Directory Express (pb.conf)

Directory Express is a basic directory lookup tool that can be used out of the box.

Figure 1-1 Directory Express: Search Result

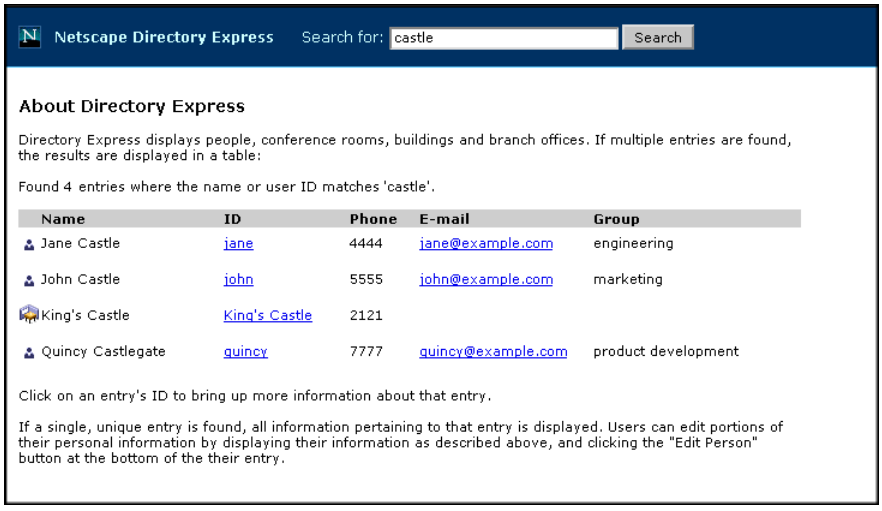
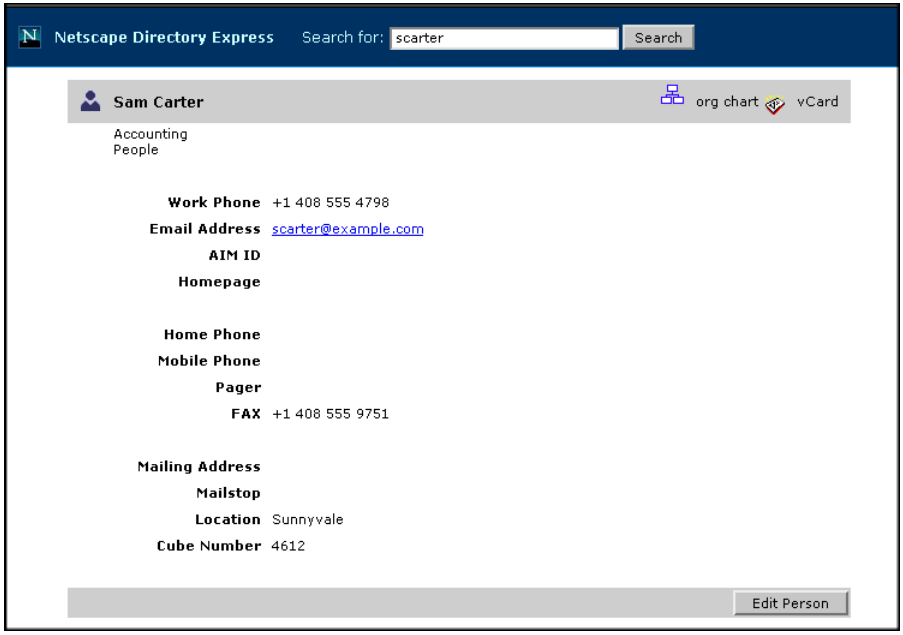

Netscape Directory Express
Search for: castle
Search
About Directory Express
Directory Express displays people, conference rooms, buildings and branch offices. If multiple entries are found, the results are displayed in a table:
Found 4 entries where the name or user ID matches 'castle'.
Name ID Phone E-mail Group
Jane Castle jane 4444 jane@example.com engineering
John Castle john 5555 john@example.com marketing
King's Castle King's Castle 2121
Quincy Castlegate quincy 7777 quincy@example.com product development
Click on an entry's ID to bring up more information about that entry.
If a single, unique entry is found, all information pertaining to that entry is displayed. Users can edit portions of their personal information by displaying their information as described above, and clicking the "Edit Person" button at the bottom of the their entry.

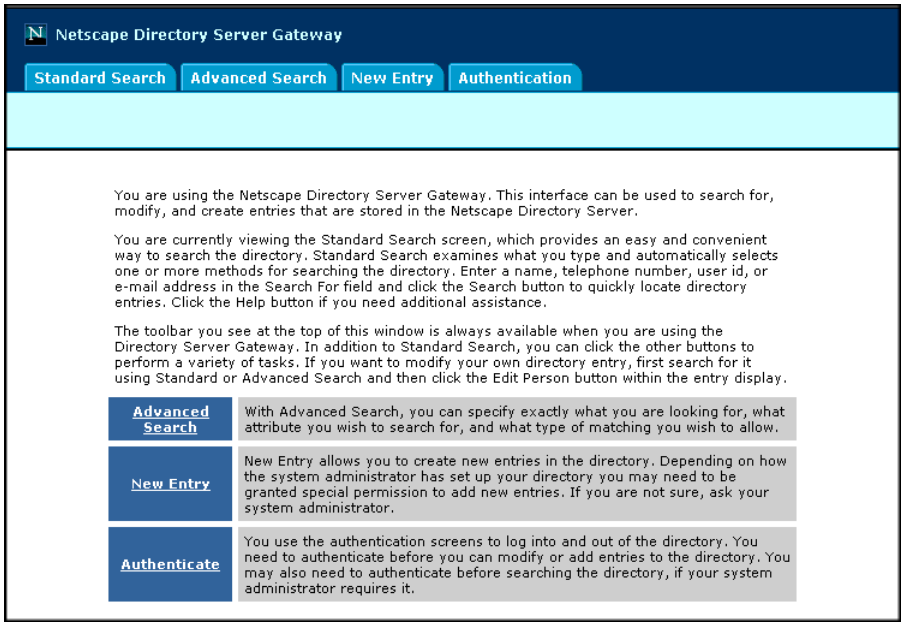
Figure 1-2 Directory Express: Extended Search Results



Default Gateway (dsgw.conf)

In addition to the standard search form, the default gateway provides an advanced search form, a Directory Server authentication form, and a form for adding and modifying entries.

Figure 1-3 Default Gateway


Netscape Directory Server Gateway
Standard Search
Advanced Search
New Entry
Authentication
You are using the Netscape Directory Server Gateway. This interface can be used to search for, modify, and create entries that are stored in the Netscape Directory Server.
You are currently viewing the Standard Search screen, which provides an easy and convenient way to search the directory. Standard Search examines what you type and automatically selects one or more methods for searching the directory. Enter a name, telephone number, user id, or e-mail address in the Search For field and click the Search button to quickly locate directory entries. Click the Help button if you need additional assistance.
The toolbar you see at the top of this window is always available when you are using the Directory Server Gateway. In addition to Standard Search, you can click the other buttons to perform a variety of tasks. If you want to modify your own directory entry, first search for it using Standard or Advanced Search and then click the Edit Person button within the entry display.
Advanced Search
With Advanced Search, you can specify exactly what you are looking for, what attribute you wish to search for, and what type of matching you wish to allow.
New Entry
New Entry allows you to create new entries in the directory. Depending on how the system administrator has set up your directory you may need to be granted special permission to add new entries. If you are not sure, ask your system administrator.
Authenticate
You use the authentication screens to log into and out of the directory. You need to authenticate before you can modify or add entries to the directory. You may also need to authenticate before searching the directory, if your system administrator requires it.

Support for Multiple Gateway Instances

Directory Server supports multiple gateway instances — that is, many gateways can access directory data from the same HTTP server without conflict.

The `.conf` files defining the configuration of gateway instances are stored in the *serverRoot*`/clients/dsgw/context` directory. Within the `.conf` file are two parameters specifying the path names for the HTML and template files for the gateway. The following lines show the HTML and configuration directories specified in the `pb.conf` file:

```
htmldir    ../pbhtml
configdir  ../pbconfig
```

Specifying Gateway Configuration to Gateway CGIs

Information about which `.conf` file to use is communicated in the `QUERY STRING` using a `GET` and through a hidden variable on a `POST`.

GET Operations (GCONTEXT)

In a `GET` operation, gateway CGIs get the gateway context from the `QUERY STRING` in the URL.

Use the `GCONTEXT` directive in all URLs to gateway CGIs. Embed

```
<!--GCONTEXT -->
```

after the CGI name, as shown in the example that follows. This directive will be replaced by the current CGI's gateway context. The `GCONTEXT` directive is the only gateway directive that does not have to be at the beginning of the line. An example of embedding the `GCONTEXT` string in a link follows:

```
<a href=/clients/dsgw/bin/lang?<!-- GCONTEXT
-->&file=auth.html>click</a>
```

POST Operations (PCONTEXT)

In a `POST` operation, the CGI posts to the gateway instance specified by a hidden variable on an HTML form. Each `POST` operation to a gateway CGI in an HTML form must use the `PCONTEXT` directive so that CGIs can pass the gateway instance to the next page and maintain the state.

For CGI invocations using a `POST`, put

```
<!-- PCONTEXT -->
```

at the beginning of a line. For example, you can specify `PCONTEXT` in an HTML form as follows:

```
<form method=post action=/clients/dsgw/bin/dosearch>
<input type=hidden name=dn value="">
<!-- PCONTEXT -->
</form>
```

For more information about content-related directives, see "Context-Related Directives," on page 109.

URLs for Gateway Access

To access a gateway, users enter the following URL:

```
http://adminHost:adminPort/clients/dsgw/bin/lang?context=<context>
```

The default gateway and Directory Express are available from the page for the Red Hat Administration Server (`http://`*adminHost*`:`*adminPort*).

Anonymous and Non-Anonymous Searching

The gateway supports both *anonymous* and *non-anonymous* searching. Anonymous searching provides basic permissions for accessing information in the user directory. A bind DN and bind password, stored in a `binddnfile`, can be set up for users to authenticate to the Directory Server. User permissions for directory access can be defined in the Red Hat Console.

If the Directory Server contains authentication credentials for a user, these override the bind DN and bind password in the gateway's `binndnfile` parameter. When authentication credentials expire or are invalid, the gateway attempts to authenticate the user to the directory using the `binddnfile`. When no `binddnfile` is specified, the gateway instance binds anonymously.

binddnfile

The location of the `binddnfile` containing bind DNs and bind passwords for individual users and groups of users is specified in the gateway's `.conf` file.

NOTE The `binddnfile` contains highly sensitive information. Do not store the `binddnfile` under *serverRoot*`/clients/dsgw` or in any directory that is served up over HTTP (for instance, `/bin/slapd/server` is a good place to store the `binddnfile`).

Automatic Updates to Directory Configuration

A script that updates gateway instances with changes to Directory Server configuration, `updatedsgw`, is included with the Directory Server installation. This script searches *serverRoot*`/clients/dsgw/context` for gateway instances that match the Directory Server host and port.

The `updatedsgw` script runs automatically for gateways installed on the Red Hat Administration Server managing the Directory Server instance. When the server port or root DN (or other settings, such as directory manager) for a Directory Server instance is changed, the Red Hat Administration Server managing the Directory Server instance runs the `updatedsgw` script.

For more information, see “Updating the Gateway with Changes to Directory Server Configuration,” on page 31.

Chapter 2

Setting Up the Gateway

This chapter describes the planning decisions and tasks required to install and initially configure a gateway for access by end users. The chapter contains the following sections:

- Gateway Installation Planning (page 29)
- HTTP Server Configuration (page 32)
- Creating a New Gateway Instance (page 36)
- Gateway .conf File Configuration (page 37)
- Configuring Gateway Clients (page 40)

Gateway Installation Planning

The following sections describe the steps for planning your installation of the gateway:

- Location of Gateway Files
- Securing Gateway Configuration and Settings
- Updating the Gateway with Changes to Directory Server Configuration
- HTTP Server Recommendations for Directory Server Gateway

Location of Gateway Files

Table 2-1 shows the locations of gateway files.

Two gateway instances are installed during Directory Server installation: Red Hat Directory Express (Directory Express) and the default gateway. The configuration files (`pb.conf` and `dsgw.conf`) for the two instances are stored in the *serverRoot*`/clients/dsgw/context` directory. Additional gateways can be created by customizing Directory Express or the default gateway.

Table 2-1 Location of Gateway Files

File Type	File Path
Default gateway configuration file	*serverRoot*`/clients/dsgw/context/dsgw.conf`
Default gateway (dsgw) HTML and template files	*serverRoot*`/clients/dsgw/html` *serverRoot*`/clients/dsgw/config`
Directory Express configuration file	*serverRoot*`/clients/dsgw/context/pb.conf`
Directory Express (pb) HTML and template files	*serverRoot*`/clients/dsgw/pbhtml` *serverRoot*`/clients/dsgw/pbconfig`

Gateway Cloning

Unique gateway instances may have unique HTML directories (for example, `..clients/dsgw/mythml`) and template directories (for example, `..clients/dsgw/myconfig`). However, gateways may also be cloned to use identical HTML and template directories while pointing to different Directory Servers or different suffixes on a Directory Server.

For more information on cloning the gateway, see “Gateway Cloning,” on page 37.

Securing Gateway Configuration and Settings

The following sections describe procedure for protecting the configuration information of your gateway.

- Protecting Bind DN and Password
- Protecting Root Processes on UNIX Systems

Protecting Bind DN and Password

The gateway configuration files reference files that contain sensitive information, including the `binddnfile` parameter containing the bind DN and bind password used to permit non-anonymous searching of the directory. The `binddnfile` should not be stored under the gateway configuration directory (*serverRoot*`/clients/dsgw`) or in any directory that is served up over HTTP.

Protecting Root Processes on UNIX Systems

On UNIX systems, it is not advisable to run the gateway from a Red Hat Administration Server that is also running a server process as `root`. This may expose sensitive information about the configuration of your Directory Servers.

Updating the Gateway with Changes to Directory Server Configuration

Directory Server Gateway includes a script, `updatedsgw`, that can be used to update all gateway instances with changes to the Directory Server configuration, including changes to Directory Server port, host, suffix, and root DN (the ability to update the suffix is not available in the server administration console). The `updatedsgw` script is stored in the *serverRoot*`/bin/slapd/admin/bin` directory.

Changes made to the Directory Server configuration (`dse.ldif`) by the Red Hat Console are posted to `updatedsgw,` and the relevant gateway files are updated. These files will be updated only when the host and port for the gateway match the host and port of the Directory Server.

NOTE	The Directory Server's root DN (the Directory Server's superuser) must match the value of the gateway's `dirmgr` parameter.

HTTP Server Recommendations for Directory Server Gateway

The Red Hat Administration Server is the default HTTP server for the two gateway clients that are installed with the Directory Server. Both Directory Express and the default gateway are preconfigured to run under the Administration Server without additional setup.

There are many factors affecting gateway performance on an HTTP server, including the following:

- The number of users accessing the gateway at a given time.
- The complexity of the directory searches performed and the search results required.
- Whether the gateway is additionally to be used for authentication and login.
- The load from other processes managed by the host machine.
- The speed and performance of the computer hardware selected for the host computer.
- The speed and capacity of the network (network hardware and software).

In general, gateway performance on the Administration Server will begin to slow down when the number of users accessing the gateway throughout the enterprise reaches 6,000 people. (This is a very general recommendation that does not take into account factors listed above, especially the speed of the host machine.)

NOTE It is not advisable to run the gateway from an Administration Server that is also running a server process as `root`. This may expose sensitive information about the configuration of your Directory Servers.

Running the Gateway in High-Usage Networks

Network administrators expecting high gateway usage may wish to move the gateway to a high-performance HTTP server that is dedicated to running the gateway.

HTTP Server Configuration

The following sections describe the steps for configuring an HTTP server:

- Name Translation Mapping
- Gateway Root Suffix
- Configuring the Gateway for Web Servers

Name Translation Mapping

The HTTP server uses Name Translation mapping to translate a virtual path provided by a gateway client to a physical path used by an HTTP server. This Name Translation mapping specifies the gateway's HTML directory. The gateway's CGIs use this information to output the correct URL (HTTP redirection). The NameTrans mapping is specified in the gateway's configuration file using the `gwnametrans` parameter.

For more information on configuring the `gwnametrans` parameter, see "gwnametrans," on page 97.

Gateway Root Suffix

Directory Express and the default gateway are set to the root suffix specified during Directory Server installation. This suffix specifies the DN for the LDAP database and represents a root in the directory tree (for example, `dc=example,dc=com`). Multiple gateways can be set up on an HTTP server that provide access to directory entries that correspond to this root suffix.

When the Directory Server's suffix changes, it is necessary to run the `updatedsgw` script manually to propagate the change to all gateway instances.

NOTE	When the root suffix, directory manager, or port change, the gateway settings in `dsgw.conf` must be updated to reflect the changes (if they haven't been updated by Red Hat Console).

Configuring the Gateway for Web Servers

Directory Express and the default gateway are installed with the Directory Server and configured to run under the Red Hat Administration Server, which is the default HTTP server for the gateway clients. No additional configuration is necessary. However, customers in high-usage networks may wish to move their gateways (or set up new gateways) on a high-performance HTTP server.

Setting up a gateway with a web server typically requires:

1. Changing all the host names and port numbers in the configuration files (`config.txt`, `dsgw.conf`, `pb.conf`, `default.conf`, and so on).

2. Adding the following CGI directories (under Program Management).

   Prefix: `/clients/dsgw/bin`
   CGI Directory: *serverRoot*`/clients/dsgw/bin`

   (On Windows, add them as shell CGI directories.)

3. Adding an additional Document directory (under Content Management).

   Prefix: `/clients`
   Directory: *serverRoot*`/clients`

4. Changing permissions of the cookie directory (required for UNIX only).

The configuration procedures outlined in this section assume that a Red Hat Enterprise Server is installed and configured to communicate with Directory Server. For Red Hat Enterprise Server documentation, check this site:

`http://www.redhat.com/docs/manuals/dir-server/`

For configuring other HTTP servers, follow the documentation that came with the product.

To configure the gateway to work with Red Hat Enterprise Server, follow the instructions below:

1. Add an additional CGI directory.

   Adding an additional CGI directory is necessary to make the gateway's CGI programs available. For additional information, see

   `http://enterprise.netscape.com/docs/enterprise/611/admin/esprgrm.htm#21309`.

   From the Class Manager for the Red Hat Enterprise Server:

   a. Select Programs > CGI Directory.

   b. In the URL Prefix field, enter the URL prefix to use:

      `/clients/dsgw/bin`

   c. In the CGI Directory field, enter this (replace *serverRoot* with your installation directory):

      serverRoot`/clients/dsgw/bin`

   d. Click OK, then Save and Apply.

2. Add an additional document directory.

   Adding an additional document directory is necessary to establish access to the gateway files. For additional information, see

   `http://enterprise.netscape.com/docs/enterprise/611/admin/escontnt.htm#22280`.

   From the Class Manager for the Red Hat Enterprise Server:

   a. Select Content Management > Additional Document Directories.

   b. In the URL Prefix field, enter this:

      ```
      clients/dsgw
      ```

   c. In the Map to Directory field, enter this (replace *serverRoot* with your installation directory):

      serverRoot`/clients/dsgw/`

   d. Click OK, then Save and Apply.

3. Change permissions of cookie directory (required for UNIX only).

   To enable the gateway to store cookies on the HTTP server, the gateway must have write access to the HTTP server's cookie directory.

   From the Class Manager for the Red Hat Enterprise Server:

   a. Select System Settings > View Server Settings, and note the value set for the User field.

      If this value is set to `nobody`, check to make sure that the server is not running as a named user. For example, on Solaris, grep for the HTTP process:

      ```
      ps -ef | grep http
      ```

      The process listed identifies the name under which the HTTP process is running.

   b. Log into the machine as `root`.

   c. Go to the *serverRoot*`/clients/dsgw` directory and enter this:

      `# chown` *uid* `authck`

      where *uid* is the user name determined in step a.

   d. Verify that the directory is accessible by opening this URL:

```
http://webserverHost:webserverPort/clients/dsgw/bin/search
```

where *webserverHost* is the HTTP server's hostname and *webserverPort* is the port number used by the server. When the HTTP server is using the standard HTTP port number (80), the port number does not need to be included in the URL.

Creating a New Gateway Instance

These instructions assume that the new gateway instance will run under the Red Hat Administration Server or a similarly capable HTTP server.

1. Rename the `dsgw.conf` or `pb.conf` file to a new gateway context.

   For example, `clients/dsgw/context/dsgw.conf` might become `clients/dsgw/context/example.conf`.

2. Set the `gwnametrans` parameter in the new gateway's `.conf` file to point to the HTML directory.

   For example, the `gwnametrans` parameter setting for `example.conf` should point to `/clients/dsgw/examplehtml`.

3. To support non-anonymous searching (one individual user DN and password per directory instance) using the new gateway, set the `binddnfile` parameter in `example.conf` to point to the location of the file containing the bind DN and bind password that will be used to access information in the user directory.

   The `binddnfile` contains sensitive information; for security purposes, do not store the `binddnfile` within the `/clients/dsgw` directory or within any directory served up over HTTP.

4. Create an HTML directory for the new gateway.

   For example, to provide an HTML directory for `example.conf`, copy and rename an existing HTML directory (`clients/dsgw/html` or `clients/dsgw/pbhtml`) to `/clients/dsgw/examplehtml`.

5. Create a template directory containing object class templates and other configuration files.

   For example, to provide a template directory for `example.conf`, copy and rename an existing template directory (`/clients/dsgw/config` or `/clients/dsgw/pbconfig`) to `/clients/dsgw/exampleconfig`.

6. Edit the `htmldir` and `configdir` parameters in `example.conf` to point to the new HTML and template directories.

7. To access the new gateway instance (in this example, `example.conf`) navigate the browser to this URL:

   `http://`*adminHost*`:`*adminPort*`/clients/dsgw/bin/lang?context=example`

Gateway Cloning

The HTML and template directories for one gateway can serve as the HTML and template directory for many others. Maintaining the functionality of multiple gateways in centralized `/config` and `/html` directories is useful when the only values that are likely to change are parameter settings in the `.conf` file, such as the host and port specified by the `baseurl` parameter, the root DN specified by the `dirmgr` parameter, and the root suffix specified by the `location-suffix` parameter.

Gateway .conf File Configuration

The following sections describe the steps for configuring the gateway `.conf` file:

- Changing the Default Port Setting
- Setting Up a Directory Manager for the Gateway
- Setting Up the Suffix for Adding Entries
- Setting Up SSL Support
- Setting vCard Properties

Changing the Default Port Setting

The LDAP port is set during Directory Server installation. This value can be changed in the `baseurl` parameter. The following example shows the syntax used to specify a port number that is different than the default port number of 389. For example, the `baseurl` parameter in the LDAP port is changed to the following:

`baseurl` “`ldaps://dirserver.example.com:3000/o%3Dexample.com`”

Setting Up a Directory Manager for the Gateway

When Directory Server is installed, a default Directory Manager account (`cn=Directory Manager`) is setup with permissions to the root DN. The Directory Server installation requires a root DN.

It is strongly recommended that you use a different directory manager account for the gateway, an account other than `cn=Directory Manager`. Once you setup the new directory manager account (for example, `cn=gateway manager,cn=config`), use ACLs to restrict access to applicable sub suffixes and the user entries under those sub suffixes. This enables the gateway directory manager to change those users' passwords but prevents the entry from having complete control of the Directory Server.

NOTE For security reasons, set the gateway Directory Manager to an entry other than `cn=Directory Manager`.

Configuring the Directory Manager DN

Use this procedure to configure the gateway Directory Manager to reference the correct DN:

1. Create an entry for the gateway Directory Manager, making sure to set a password for the entry.
2. Set the permissions for the Directory Manager so that it has read and write authority for the entries it will manage.
3. When necessary, change the `dirmgr` parameter to refer to the Directory Manager's distinguished name (DN).

NOTE End users frequently forget their passwords, so give the gateway Directory Manager write access to the `userPassword` attribute for the entries it will manage.

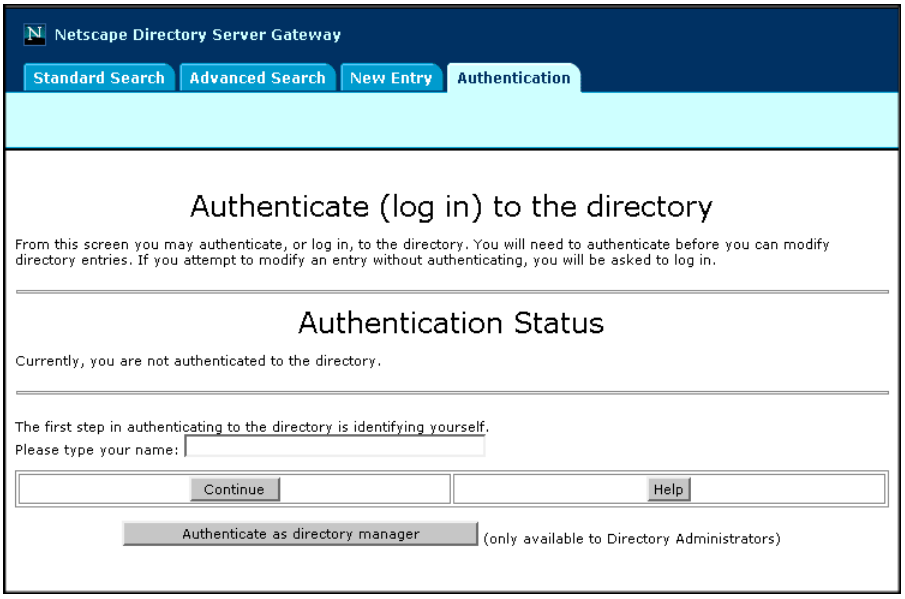
The `dirmgr` parameter is described in "dirmgr," on page 95. Creating directory entries is described in the *Red Hat Directory Server Administrator's Guide*.

Authenticating as Directory Manager

Figure 2-1 shows the authentication login screen for the default gateway. Administrators can use it to authenticate as the Directory Manager. The Authenticate as Directory Manager button is displayed only when a Directory Manager has been configured for the gateway.

The `authlifetime` parameter, which defines the number of seconds that a user may remain authenticated, is described in "location," on page 98.

Figure 2-1 Authenticating as Directory Manager


Netscape Directory Server Gateway
Standard Search
Advanced Search
New Entry
Authentication
Authenticate (log in) to the directory
From this screen you may authenticate, or log in, to the directory. You will need to authenticate before you can modify directory entries. If you attempt to modify an entry without authenticating, you will be asked to log in.
Authentication Status
Currently, you are not authenticated to the directory.
The first step in authenticating to the directory is identifying yourself.
Please type your name:
Continue
Help
Authenticate as directory manager
(only available to Directory Administrators)

Setting Up the Suffix for Adding Entries

The `location-suffix` parameter is defined in `dsgw.conf` and identifies the suffix under which the gateway creates new entries in the directory. The `location-suffix` parameter can point to any suffix in a directory.

Setting the `location-suffix` parameter is described in “include,” on page 98. The *Red Hat Directory Server Administrator’s Guide* describes the `Suffix` parameter and provides syntax examples. Setting the root suffix is also described in the *Red Hat Directory Server Installation Guide*.

Setting Up SSL Support

When the Directory Server is installed, the gateway is configured to communicate with the Directory Server using a non-SSL host name and port number. This information is stored in the `baseurl` parameter.

Configuring the gateway to use SSL when communicating with the Directory Server requires modification of the `securitypath` and `baseurl` parameters in `dsgw.conf`.

Enabling SSL communications on the Directory Server is described in the *Red Hat Directory Server Administrator's Guide*. Information about managing key and certificate databases is provided in *Managing Servers with Red Hat Console*.

Configuring the Gateway to Use SSL

The `securitypath` parameter specifies the location of the certificate database. For example, you can specify the path to the certificate database as follows:

```
securitypath "/usr/redhat/servers/alias/slapd-testDir-cert8.db"
```

The following example shows the `baseurl` parameter configured to use `ldaps` (instead of `ldap`, the default) and standard SSL port number 636:

```
baseurl "ldaps://dirserver.example.com:636/o%3Dexample.com"
```

NOTE Before configuring SSL, verify that the gateway's certificate database contains a server certificate or Certificate Authority (CA) certificate needed to communicate with the Directory Server.

For more information about the `baseurl` parameter, see "baseurl," on page 92.

Setting vCard Properties

Mappings between vCARD properties and LDAP attribute type are described in "vcard-property," on page 103.

Configuring Gateway Clients

The following sections describe how to configure clients of the gateway:

- Language Support for HTTP Clients
- Displaying a Non-English Alphabet
- Configuring Netscape 7.x for Preferred Language
- Customizing Communicator's LDAP Settings

Language Support for HTTP Clients

When a user accesses information in the directory from an HTTP client — through the gateway or another HTTP-based LDAP interface — the client provides the Directory Server with information indicating the optimal character set and collation order to use in transmitting information to the browser.

Unicode and Latin-1 Character Sets

When the user is using Netscape Communicator, the Directory Server sends Unicode characters.

Displaying a Non-English Alphabet

To display directory content that uses a non-English alphabet, a font capable of displaying a non-English alphabet must be installed on the user's system.

The Directory Server can store any Unicode character, so users of Netscape Communicator should install a font that supports all of Unicode. Bitstream Cyberbit, which is bundled with Communicator, supports Unicode.

Users who are not using Communicator should use a font that supports Latin-1 (or Western) character sets. Most of the commonly used fonts (Courier, Times Roman, Helvetica) have a Latin-1 variant.

Configuring Netscape 7.x for Preferred Language

1. Install a font that supports Unicode.
2. In the browser window, go to Edit > Preferences > Appearance > Fonts.
3. From the Fonts For pull-down menu, select Unicode.
4. Set the appropriate font type, size, and display resolution.
5. Go to Edit > Preferences > Navigator > Languages/Content, and configure the list of languages so that the best description of the user's language is first, followed by other acceptable languages.

   For example, a speaker of British English who also reads Spanish might list English/United Kingdom [en-GB] first, followed by English [en], and then Spanish [es].

Customizing Communicator’s LDAP Settings

Administrators can reconfigure Javascript preference settings in Communicator to allow users to interact with information stored in the user directory.

- In the Address Book and Select Address dialog boxes (accessible from the mail composition window), users can enter one string of search criteria to search an LDAP directory for matching names.
- In the Search Directory dialog, users can enter more complex query expressions to search an LDAP directory using native LDAP searches.
- Users can enter LDAP URLs (beginning with the “`ldap://`” prefix) in Navigator (web browser) windows to search an LDAP directory.

Chapter 3

Gateway Localization

This chapter describes gateway localization and identifies the tasks required to set up additional gateway locales. The chapter contains the following sections:

- Unicode and Support for UTF-8 (page 43)
- How the Gateway Selects a Character Set (page 44)
- Special Characters (page 45)
- Gateway Locales (page 46)
- Setting Up Locales for Translation (page 46)

Unicode and Support for UTF-8

Unicode is a character set containing all the characters of all the world's major languages. There are different standard methods to encode Unicode, including UCS-2, which is NT's Unicode version, and UTF-8, the version of Unicode specified by version 3 of the LDAP protocol.

The Directory Server and associated applications use UTF-8 in versions 2 and 3 of LDAP. Most software included in the Directory Server uses UTF-8 internally and at interfaces other than LDAP (for example, in command-line parameters and LDIF files).

NOTE Netscape Communicator and Netscape browsers support UTF-8.

How the Gateway Selects a Character Set

The gateway can output web pages in many character sets. The gateway selects a character set for each HTTP client based on a combination of input from the client and from the gateway's configuration files. The gateway selects a character set for transmission according to this priority:

- Character set defined in the client's `HTTP Accept-charset` header. (This can be overridden for a particular browser using the `ignoreAcceptCharsetFrom` parameter).
- Character set defined in the client's `HTTP Accept-language` header. (For example, for Japanese, the charset would be defined as `..clients/dsgw/ja/dsgwcharset.conf`).
- Character set defined in the gateway's `.conf` file by the charset parameter.

How the Gateway Selects from Multiple Requested Characters Sets

When a client includes more than one character set in a request header, and the gateway supports more than one of these, it selects a character set according to this priority:

- UTF-8
- Of the possible character sets, the character set with the highest Q value (for example, `de;q=1, en;q=0.5, fr;q=0.7` would give German the highest Q value)
- The character set that appears first in the request header.
- Latin-1 (ISO-8859-1)

HTTP Clients that Request UTF-8

Browsers designed for localization are configured to request the UTF-8 character set by default. To support localization, the gateway is preconfigured to transmit the UTF-8 character set to these clients: Netscape Communicator and Internet Explorer. The gateway allows this preconfiguration to be overridden using the `ignoreAcceptCharsetFrom` parameter. For more information about this parameter, see “ignoreAcceptCharsetFrom,” on page 97.

The conversion from UTF-8 to the gateway client's chosen character set is performed shortly before output.

HTTP Clients that Do Not Request UTF-8

For browsers that do not request UTF-8 by default, the gateway selects a character set from the `Accept-Charset` request header or from the `Accept-Language` request header, depending on the HTTP client.

Some HTTP clients don't request any character set information. For these clients, the gateway's charset parameter definition is the default. When the charset parameter is not defined in the `dsgw.conf` file, the gateway uses Latin-1 (which is the default in HTTP).

In addition to UTF-8 and Latin-1, the gateway can convert to and from several national character sets, depending on the client's needs and configuration, including the following:

- Shift_JIS
- Big5
- EUC-KR

Special Characters

The following sections describe how special characters are interpreted by the gateway:

- Non-Breaking Space
- Query Strings

Non-Breaking Space

If the client's character set lacks a character for non-breaking space, but has ideographic space, non-breaking spaces are converted to ideographic spaces before charset conversion.

See the `changeHTML` directive (page 93) in the gateway configuration file `dsgw.conf`.

Query Strings

When the gateway needs to embed a UTF-8 string in a URL, it encodes it in a query string (the query string is the part of the URL that follows the question mark).

This works around a problem with Japanese NT, which garbles environment variables that are in UTF-8 (or any charset except `Shift_JIS`). The Web server passes information to the gateway CGI programs in environment variables, but the query string environment variable `$QUERY_STRING` is URL-encoded, so it can handle UTF-8 (from Windows' point of view, it's ASCII).

Gateway Locales

The gateway's default language is US English.

Support for Multiple Locales

A single gateway instance supports clients in multiple locales concurrently.

Support for multiple locales is accomplished by translating documentation (including online help), the string resource database, and the configuration and HTML template files. A single copy of the compiled code handles all supported locales.

Locale-dependent information is stored in translated files stored in subdirectories identifying the locale name. These editable files are stored separately from the gateway code. For example, the German translation of `config/search.html` is stored in `config/de/search.html`, the French translation is stored in `config/fr/search.html`, and the Japanese translation is stored in `config/ja/search.html`.

Setting Up Locales for Translation

The default gateway can be configured to support locales in addition to English (the default locale), French, German, Spanish, and Japanese. This is part of the overall localization effort, which includes localizing all the configuration and HTML files, including the online help and the string resource database. This is made possible by including a pointer to the mapping table in the `dsgw-l10n.conf` file, which is stored during Directory Server installation in the `lang` directory:

serverRoot`/clients/dsgw/config/lang`

dsgw-l10n.conf

`dsgw-l10n.conf` provides translation in the Search and Advanced Search pull-down menus for the default gateway (`dsgw.conf`). If `dsgw-l10n.conf` is not present in the `/config/lang` directory, translation of the UI does not occur and English characters appear in the pull-down menus for Standard Search and Advanced Search.

The following example shows how to create a new locale using Chinese as the language for translation:

1. Create a `zh` directory in the *serverRoot*`/clients/dsgw/context` directory.
2. Copy the dsgw.conf file to the *serverRoot*`/clients/dsgw/context/zh` directory.
3. Open the gateway's `.conf` file in a text editor, and uncomment this line from the file:

   ```
   include "../config/dsgw-l10n.conf"
   ```

4. Save your changes, and close the file.
5. Create a zh directory in the serverRoot/clients/dsgw/config directory.
6. Copy (or create) the `dsgw-l10n.conf` file (stored during gateway installation in *serverRoot*`/clients/dsgw/config/lang`) to the *serverRoot*`/clients/dsgw/config/zh` directory.

NOTE If you are using the US version of the gateway, `dsgw.conf` contains a sample of `dsgw-l10n.conf`.

Chapter 4

File Controlling Gateway Functionality

This chapter provides examples of customized gateways. The chapter contains the following sections:

- Files Controlling Gateway Functionality (page 49)
- Gateway .conf Files (page 51)
- Gateway Search Configuration Files (page 51)
- Object Class Templates (page 52)
- Gateway Script Files (page 53)
- Gateway Search Result Templates (page 53)
- Banner Files (page 54)

Files Controlling Gateway Functionality

To the gateway user, the gateway is a set of HTML forms that can be accessed from a web browser to communicate directly with the Directory Server over HTTP.

To the gateway administrator, the gateway is controlled by a set of files installed during Directory Server installation. These files can be edited to:

- Create new gateway instances.
- Edit the object class attributes that define the entries users can add to the Directory.
- Edit the search object class attributes that define how people search for and view entries in the LDAP directory.
- Change the appearance of gateway forms.
- Update the gateway with changes to Directory Server configuration.

Files that control gateway functionality are described in Table 4-1.

Table 4-1 Gateway File Types and Locations

Files	Function	Location
Gateway `.conf` files	Define basic configuration parameters for the gateway and specify the HTML and template directory. (See "Gateway .conf Files" on page 51.)	*serverRoot*`/clients/dsgw/context`
Gateway search configuration files	Define how the gateway performs searches in the Directory. (See "Gateway Search Configuration Files" on page 51.)	*serverRoot*`/clients/dsgw/config`
Object class templates	Define object classes and attributes for entry types. (See "Object Class Templates" on page 52.)	*serverRoot*`/clients/dsgw/config`
Gateway script files	Contain scripts used to communicate instructions to the Directory Server over HTTP. (See "Gateway Script Files" on page 53.)	*serverRoot*`/clients/dsgw/config`
Gateway search result templates	Define the filters used to display search results to the user. (See "Gateway Search Result Templates" on page 53.)	*serverRoot*`/clients/dsgw/config`
Banner files	Define appearance of colors, background, graphics on gateway forms. (See "Banner Files" on page 54.)	*serverRoot*`/clients/dsgw/html`

Gateway .conf Files

A gateway's configuration file (`.conf` file) describes an instance of the gateway. The file controls the host, port, root suffix, and Directory Manager when communicating with the Directory Server. The file also controls:

- The locations where new entries can be created within the directory.
- The types of entries that can be created.
- The search base.
- Whether the gateway uses SSL communications.

dsgw.conf

`dsgw.conf` is the configuration file for the default gateway. `dsgw.conf` is invoked at:

`http://`*adminHost*`:`*adminPort*`/clients/dsgw/bin/lang?context=dsgw`

pb.conf

`pb.conf` is the configuration file for Directory Express, a gateway customized as a directory lookup tool. `pb.conf` is invoked at:

`http://`*adminHost*`:`*adminPort*`/clients/dsgw/bin/search?context=pb`

Gateway configuration parameters are defined in Appendix A, “Parameters Defined in the .conf File.”

Gateway Search Configuration Files

Gateway search configuration files determine how the gateway queries information in the directory and returns search results to the users. Gateway search configuration files are stored in the *serverRoot*`/clients/dsgw/config` directory. All gateway instances for a Directory Server use these common search configuration files.

- `dsgwsearchprefs.conf`

  This file specifies the object classes and object class attributes than can be searched.

- `dsgwfilter.conf`

  This file specifies the search filters used to search for a particular object class. The gateway uses this file when performing a standard search operation.

The `dsgwsearchprefs.conf` and `dsgwfilter.conf` files are discussed in Chapter 6, “Search Attributes, Filters, and Results.”

Object Class Templates

The gateway contains a template file for each object class defined in the gateway. To modify how the gateway displays an entry type, edit the corresponding template file. To add gateway support for a new object class, create a new template file, or modify an existing one.

Modifying template files is discussed in Chapter 5, “Editing Entry Types.”

Default gateway object class templates stored in the *serverRoot*/`clients/dsgw/config` directory are listed in Table 4-2.

Table 4-2 Default Template Files and Related Object Classes

Template File	Related Object Class
`display-dc.html`	domain
`display-country.html`	country
`display-group.html`	groupOfNames
`display-groupun.html`	groupOfUniqueNames
`display-ntgroup.html`	ntGroup
`display-ntperson.html`	ntUser
`display-org.html`	organization
`display-orgperperson.html`	inetOrgPerson
`display-orgunit.html`	organizationalUnit
`display-person.html`	person

Gateway Script Files

Script files are used to generate HTML forms dynamically for the user, based on information supplied by the user and information retrieved from the Directory Server. Script files contain directives the gateway uses to construct the HTML for a page.

Script files can be modified to change the appearance of text that appears in fields, buttons, and prompts on gateway forms.

Script files used to modify the information provided on gateway forms are stored in the *serverRoot*/`clients/dsgw/config` directory and are identified in Table 4-3.

Table 4-3 Gateway Forms and Corresponding Script Files

Set of Forms	Corresponding Script Files
Authentication forms	`authPassword.html` `authSearch.html`
Standard search forms	`searchString.html`

Advanced search forms	`csearchAttr.html`
	`csearchBase.html`
	`csearchMatch.html`
	`csearchString.html`
	`csearchType.html`

Newentry forms	`newentry.html`
	`newentryType.html`
	`newentryName.html`

Gateway Search Result Templates

Search result templates control how the results of a standard or advanced search are displayed when more than one entry is found. A separate search result file exists for each search object defined in `dsgwsearchprefs.conf`.

The default search result files installed in the *serverRoot*`/clients/dsgw/config` directory during Directory Server installation are shown in Table 4-4.

Table 4-4 Default Search Result Templates

Search Result Template	Related Object Class
`list-Anything.html`	Anything
`list-Auth.html`	Auth (internal)
`listDomaincomponent.html`	Domain
`list-Groups.html`	Groups
`list-NT-Groups.html`	NT-Groups
`list-NT-People.html`	NT-People
`list-Org-Units.html`	Org-Units
`list-Organizations.html`	Organization
`list-People.html`	People

Chapter 6, “Search Attributes, Filters, and Results,” describes how search result templates can be edited to modify the display of search results.

Banner Files

Banner files identified in Table 4-5 are used to specify the banner and button images that appear in gateway forms.

Table 4-5 Banner Files Controlling Appearance of Gateway Forms

Banner File	Controls Appearance of Banner and Buttons in ...
`maintitle.html`	Main form
`authtitle.html`	Authentication form
`csearchtitle.html`	Advanced search forms
`newentrytitle.html`	New entry forms
`searchtitle.html`	Standard search form
`display-*.html`	View or edit entry forms

Chapter 7, “Customizing Graphics and Color,” describes how to alter the color schemes and images appearing on gateway forms.

Chapter 5

Editing Entry Types

This chapter describes how entry type formats — defined by object classes and their attributes — can be controlled by editing parameters in the `dsgw.conf` file. The chapter contains the following sections:

- Entry Types (Object Classes) (page 55)
- Mapping Locations and Entry Types (page 57)
- Object Class Attributes in Template Files (page 59)

Entry Types (Object Classes)

The following sections describe entry types in detail:

- Parameters Controlling Entry Types
- Considerations for Adding New Entry Types

Parameters Controlling Entry Types

The functionality of entry types appearing on gateway forms are controlled by parameters stored in `dsgw.conf`:

- Template parameter settings determine the types of objects that can be created and the attributes supported for object classes.
- Newtype parameter settings determine the DN formats to be applied to new entries.
- Location parameter settings determine where in the directory new entries reside.

Parameters in the `dsgw.conf` file are described in Appendix A, “Parameters Defined in the .conf File.”

template

The `template` parameter is used to map the gateway's HTML templates for entry types to the Directory Server's LDAP object classes.

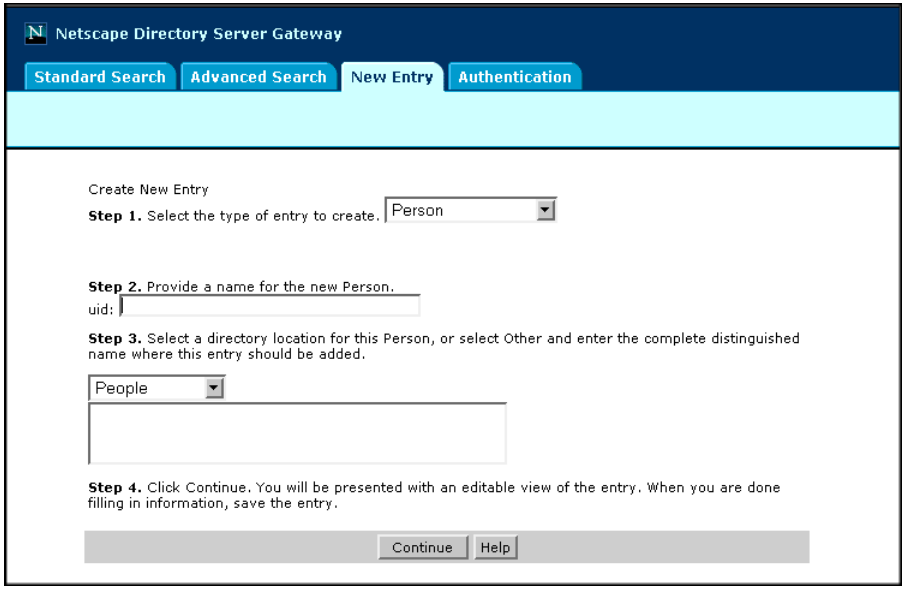
location

The `location` parameter is used to define points in the directory tree where new entries can be added. The `location` parameter definitions must precede `newtype` parameter definitions in the `.conf` file.

newtype

Each entry type is described by a `newtype` parameter. The newtype template indicates how the entry will be formatted and the location in the directory tree where the entry will be created. Entry types for the default gateway, such as the Create New Entry form (Figure 5-1), appear in the pop-up menu gateway forms,.

Figure 5-1 New Entry Form


Netscape Directory Server Gateway
Standard Search
Advanced Search
New Entry
Authentication
Create New Entry
Step 1. Select the type of entry to create.
Person
Step 2. Provide a name for the new Person.
uid:
Step 3. Select a directory location for this Person, or select Other and enter the complete distinguished name where this entry should be added.
People
Step 4. Click Continue. You will be presented with an editable view of the entry. When you are done filling in information, save the entry.
Continue
Help

Considerations for Adding New Entry Types

Before adding support for a new entry type (object class), decide:

- Where will the new entry be created?

  If a `location` parameter is not set up to point to the location where the new entry type will be created, add a new `location` parameter to the `dsgw.conf` file.

- How will the new entry be formatted?

  When the new object class has many attributes in common with an existing entry type, update the corresponding template line in `dsgw.conf` to support the new object class. When a new object class requires a new template, add a new `template` parameter to `dsgw.conf`.

The `newtype` and `location` parameters are described in Appendix A, “Parameters Defined in the .conf File.”

Mapping Locations and Entry Types

The `location` parameter is used to define points in the directory tree where new entries can be added. The default locations defined in `dsgw.conf` are intended for the sample directory shipped with the Directory Server. They are unlikely to match the structure of the actual directory.

This section explains the following:

- Mapping Entry Types to Locations
- Configuring DN Formats for Entry Types

NOTE	For simple directory structures, define locations that represent branch points in the directory. For complex structures, define branch points for the most commonly used directory branches only.

Mapping Entry Types to Locations

Each entry type must be mapped to a location where that type of entry can be placed. The following example shows a mapping of locations and newtype parameters in `dsgw.conf`:

```
location      country      "United States"           "c=US#"
location      org          "This Organization"       ""
location      groups       "Organizational Groups"   "ou=Groups"
location      acct         "Accounting"              "ou=Accounting"
location      hr           "Human Resources"         "ou=Human Resources"
location      pay          "Payroll"                 "ou=Payroll"
location      pd           "Product Development"     "ou=Product Development"
location      test         "Product Testing"         "ou=Product Testing"

newtype       orgperson    "Person"                  cn acct hr pay pd test
newtype       NTperson     "NT Person"               cn acct hr pay pd test
newtype       groupun      "Group"                   cn groups
newtype       orgunit      "Organizational Unit"     ou org
newtype       org          "Organization"            o country
```

In the example, locations defined in the `newtype` parameter (such as `country`, `org`, or `groups`) correspond to handles defined in the `location` parameter. The friendly names (in quotes) in the third column indicate the choices that will appear in pull-down menus on gateway forms.

NOTE `Location` parameter definitions in `dsgw.conf` must be listed before `newtype` parameter definitions.

See “Entry Types (Object Classes),” on page 55, for more information.

Setting Up Organizational Units

Assuming that the root DN is set to `o=example.com`, the mappings in the following example can be used to create people in the following organizational units:

```
ou=Accounting, o=example.com
ou=Human Resources, o=example.com
ou=Payroll, o=example.com
ou=Product Development, o=example.com
ou=Product Testing, o=example.com
```

Configuring DN Formats for Entry Types

The following sections describe how to configure entry type DNs depending upon the format.

UID-Based DN

When a person or Windows person entry is added to the directory, the gateway prompts for a unique DN. The unique DN is typically the user ID of a person in the organization.

Although DN formats can be based on the common names of employees in the organization, common names are frequently not unique within an organization.

NOTE	UID-based DN formats are recommended because they are by nature unique and can prevent naming collisions within the directory.

Modifying the Default DN Format

The default DN format can be modified by editing the `rdnattr` variable within the `newtype` parameter.

To change the gateway configuration so that person entries are created using common name-based DNs rather than user ID-based DNs, edit the following line in the `dsgw.conf` file:

```
newtype orgperson “Person” uid people special
```

to read as follows:

```
newtype orgperson “Person” cn people special
```

Object Class Attributes in Template Files

The following sections describe the object classes and attributes contained by the template files:

- Default Gateway Object Classes
- Templates and Directives
- Adding Attributes to Object Classes
- Deleting Attributes from Object Classes
- Extending Object Classes
- Creating a New Parent Object Class

Default Gateway Object Classes

The default gateway supports the object classes listed in Table 5-1.

Table 5-1 Default Object Classes

Default Gateway Object Class	Related Template File
domain component	`display-dc.html`
groupOfNames	`display-group.html`
groupOfUniqueNames	`display-groupun.html`
ntGroup	`display-ntgroup.html`
organization	`display-org.html`
organizationalUnit	`display-orgunit.html`
person	`display-person.html`
inetOrgPerson	`display-orgperson.html`
ntUser	`display-ntuser.html`
country	`display-country.html`

Templates and Directives

Object class attributes associated with an entry type are defined by *directives* contained in gateway template files. Directives are instructions, written as HTML comments, that are interpreted by the gateway's CGI scripts. Each directive is an independent, single line of HTML in a template file (with the exception of `<!-- GCONTEXT -->`, which is embedded within an URL).

Entry-Related Directives

Entry-related directives are responsible for how the gateway displays, edits, adds, and lists directory entries. The most commonly used entry-related directive is `DS_ATTRIBUTE`, which determines how attributes in LDAP entries are displayed on gateway forms.

`DS_ATTRIBUTE` directives begin with a `DS_ENTRYBEGIN` tag and close with a `DS_ENTRYEND` tag.

Appendix B, "Gateway Directives," lists the possible arguments for the `DS_ATTRIBUTE` directive.

Adding Attributes to Object Classes

Adding an attribute to an object class requires adding an additional row to the HTML table in the template file where the object class is defined.

The syntax in the following example defines an `Initials` attribute for the `orgperson` object class:

```
<TR>
<TD VALIGN= “TOP” NOWRAP>Initials</TD>

<TD VALIGN= “TOP” NOWRAP><B>
<!-- DS ATTRIBUTE “attr=Initials” “syntax=cis” “cols=>16” -->
</B></TD>
<TD> </TD>
<TD> </TD>
</TR>
```

To complete the row, two null cells are added. This maintains the HTML table format. For Asian character sets, substitute an ideographic space for the non-breaking space ( ) shown in the example.

NOTE	Attribute values are added in pairs. When adding a single attribute to an object class, remember to complete the table row.

Deleting Attributes from Object Classes

Deleting an attribute from an object class requires deleting a complete row or part of a row from the HTML table where the object is defined. The following example shows the steps required to delete the mobile phone attribute from the `orgperson` object class:

1. Open `display-orgperson.html` template, and delete the mobile phone attribute-value pair.

```
<TR>
<TD VALIGN=“TOP” NOWRAP>Pager:</TD>
<TD VALIGN=“TOP” NOWRAP><B>
<!-- DS ATTRIBUTE “attr=pager” “syntax=tel” “cols=>16” -->
</B></TD>
<TD VALIGN=“TOP” NOWRAP>Mobile Phone:</TD>
<TD VALIGN=“TOP” NOWRAP><B>
<!-- DS ATTRIBUTE “attr=mobile” “syntax=tel” “cols=>16” -->
</B></TD>
</TR>
```

2. Insert a null pair to replace the deleted attribute.

```
<TR>
<TD VALIGN=“TOP” NOWRAP>Pager:</TD>
<TD VALIGN=“TOP” NOWRAP><B>
<!-- DS ATTRIBUTE “attr=pager” “syntax=tel” “cols=>16” -->
</B></TD>
<TD> </TD>
<TD> </TD>
</TR>
```

NOTE When deleting a single attribute-value pair from a row, replace the two deleted cells with two cells containing the non-breaking space character. This maintains the table width and ensures that the background colors are rendered correctly.

Extending Object Classes

The gateway can be extended to support additional object classes. This requires changing information in an existing object class template so that the gateway displays the associated entry type.

NOTE When extending object class definitions, the child should appear below the parent object class in the HTML file. Otherwise, the gateway cannot correctly interpret the HTML syntax.

Adding a Template for a Child of a Parent Class

The easiest way to create a new object class is to extend an existing object class template, adding and deleting attributes as necessary. The following example shows the steps required to add a template for a new object class, `examplePerson`. The new template adds two custom attributes, `dateOfBirth` and `preferredOS`, to the `inetOrgPerson` object class.

1. Copy the `display-orgperson.html` file, and rename it as `display-exampleperson.html`.

2. Edit the third line in the template file to indicate the name of the new directory entry type. Change:

```
<!-- inet. organizational person directory entry -->
```

to

```
<!-- example person directory entry -->
```

3. Edit the DS_OBJECTCLASS directive to include the new object class. Change:

```
<!-- DS_OBJECTCLASS "value=person,inetOrgPerson" -->
```

to

```
<!-- DS_OBJECTCLASS "value=person,inetOrgPerson,
exampleperson" -->
```

4. Add the value for exampleperson to the input type.

```
<!-- IF "Adding" -->
<INPUT TYPE="hidden" NAME="add_objectClass" VALUE="top">
<INPUT TYPE="hidden" NAME="add_objectClass" VALUE="person">
<INPUT TYPE="hidden" NAME="add_objectClass"
VALUE="organizationalPerson">
<INPUT TYPE="hidden" NAME="add_objectClass"
VALUE="inetOrgPerson">
<INPUT TYPE="hidden" NAME="add_objectClass"
VALUE="exampleperson">
```

Be sure to add the line immediately before the ENDIF directive:

```
<!-- ENDIF // Adding -->
```

5. Add a new table row containing the dateOfBirth and preferredOS attribute-value pairs.

```
<TR>
<TD VALIGN="TOP">Date of Birth:</TD>
<TD VALIGN="TOP"><B>
<!-- DS_ATTRIBUTE "attr=dateOfBirth" "cols=>8" -->
</B></TD>
<TD VALIGN="TOP">Preferred OS</TD>
<TD VALIGN="TOP"><B>
<!-- DS_ATTRIBUTE "attr=preferredOS" "cols=>6" -->
</B></TD></TR>
```

For more information on adding attributes, see “Object Class Attributes in Template Files,” on page 59.

6. Define a `template` parameter in `dsgw.conf` for the object class `examplePerson`:

   ```
   template exampleperson person inetorgperson exampleperson
   ```

   This will instruct the gateway to display the `exampleperson` entry type according to the template defined for the `examplePerson` object class (`display-exampleperson.html`).

7. Update the Directory Server schema to include the `examplePerson` object class.

8. To allow users to add entries for `exampleperson` using the gateway, add an additional `newtype` parameter to the `dsgw.conf` file. If this entry type is for display purposes only, no `newtype` parameter needs to be added.

See “Considerations for Adding New Entry Types,” on page 57, and “Extending Search Preferences,” on page 74.

Creating a New Parent Object Class

These steps are required when the object class is not a child of an existing object class.

1. Add a template parameter to `dsgw.conf` for the new object class.

   `template` *newobjectclass*

   This will instruct the gateway to display the associated entry type according to the template defined for the new object class.

2. To allow gateway users to add entries for the entry type, add an additional `newtype` parameter to the `dsgw.conf` file. If the associated entry type is for display purposes only, no `newtype` parameter needs to be added.

3. Update the Directory Server schema to include the new object class.

4. Add a search object entry to `dsgwsearchprefs.conf`, and update `dsgwfilter.conf` so that the gateway will search for entries of this type.

5. Create a new search results form defining how the gateway will display search results for the new object class.

NOTE Modify an existing search result form to create a new search results form. See “Adding Information to Search Results,” on page 80, and “Removing Information from Search Results,” on page 81.

Chapter 6

Search Attributes, Filters, and Results

This chapter describes the files that control how the gateway searches for objects and describes how to add search support for a new object. The chapter contains the following sections:

- Search Configuration Files (page 67)
- Changing Search Scope (page 68)
- Modifying Search Attributes for Advanced Searches (page 69)
- Adding Search Support for a New Object (page 74)
- Modifying Default Search Filters (page 76)
- Customizing Search Result Templates (page 78)

Search Configuration Files

The `dsgwsearchprefs.conf` and `dsgwfilter.conf` files are the search configuration files that control the gateway's search functionality. These files are stored in the gateway's template directory (`/config` for the default gateway).

dsgwsearchprefs.conf

The `dsgwsearchprefs.conf` file specifies the preferences for searching object classes defined in the gateway. Each entry contains:

- The scope of the search within the Directory Server.
- The search filter to append to the search string (corresponding to the search filter entry defined in `dsgwfilter.conf`).

- The label of the search attribute as it is displayed in the Find drop-down list on the Search form .
- The object class attribute to search on.
- Match types to use in search results.

NOTE Define new search preferences in `dsgwsearchprefs.conf` whenever a new object class with searchable attributes is added to the gateway.

dsgwfilter.conf

The `dsgwfilter.conf` file contains an entry for each search object defined in `dsgwsearchprefs.conf`. Each entry defines the following:

- Pattern for which to search.
- Delimiters for the search pattern.
- LDAP filter for generating search results.
- Text to use in describing search results for the pattern.
- Scope of the search (not required).

The name of the search filter entry for a search object is identified in `dsgwsearchprefs.conf`.

Changing Search Scope

Search object entries in `dsgwsearchprefs.conf` define the search scope used in searches for the corresponding object class. The default scope for gateway search objects (subtree) specifies the `baseurl` and all its children.

The scope of a search object can be changed by editing the corresponding line in `dsgwsearchprefs.conf`. Valid search scopes are shown in Table 6-1.

Table 6-1 Valid Search Scopes

Search Scope	Tells the Gateway to...
`base`	Search the Directory Server for the entry specified in the `baseurl` parameter but not to search in children of the entry.

Table 6-1 Valid Search Scopes *(Continued)*

Search Scope	Tells the Gateway to...
`onelevel`	Not to search in the entry specified in the `baseurl` parameter but search in the most immediate children of the entry.
`subtree`	Search the entry specified in the `baseurl` parameter and all of its children. This is the default setting.

Modifying Search Attributes for Advanced Searches

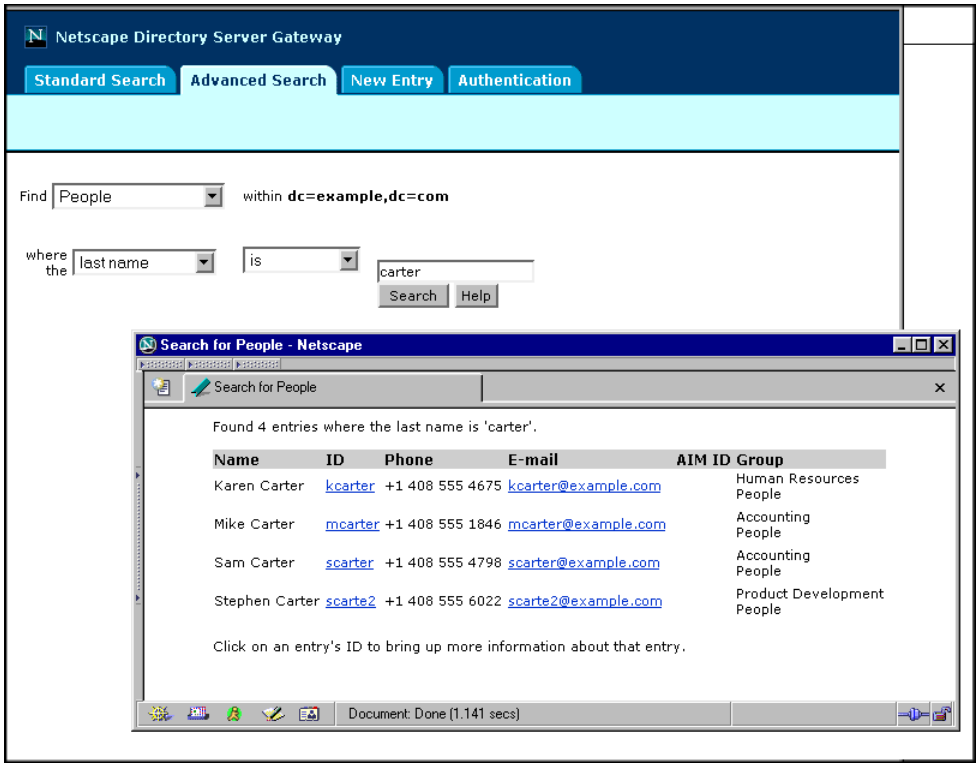
Each search object entry in `dsgwsearchprefs.conf` has a list of attributes that can be modified for advanced searches. This section explains the following:

- Standard and Advanced Searches
- Specifying Search Attributes for Person
- Directory Express Search Support for User ID
- Adding Search Support for Additional Attributes

Standard and Advanced Searches

An advanced search differs from a standard search in that users are provided with a pull-down menu of search types. In the default gateway, the Standard Search form searches on object classes defined for the gateway. The Advanced Search form allows users to also search in specific object class attributes and to specify a matching pattern. Figure 6-1 shows the Advanced Search form with search results.

Figure 6-1 Advanced Search Form: Search Results


Netscape Directory Server Gateway
Standard Search
Advanced Search
New Entry
Authentication
Find People within dc=example,dc=com
where the last name is carter
Search Help
Search for People - Netscape
Search for People
Found 4 entries where the last name is 'carter'.
Name ID Phone E-mail AIM ID Group
Karen Carter kcarter +1 408 555 4675 kcarter@example.com Human Resources People
Mike Carter mcarter +1 408 555 1846 mcarter@example.com Accounting People
Sam Carter scarter +1 408 555 4798 scarter@example.com Accounting People
Stephen Carter scarte2 +1 408 555 6022 scarte2@example.com Product Development People
Click on an entry's ID to bring up more information about that entry.
Document: Done (1.141 secs)

The figures that follow show the matching patterns that can be selected in the Advanced Search form.

Figure 6-2 shows the entry type pop-up menu on the Advanced Search form.

Figure 6-2 Advanced Search Form: Entry Type

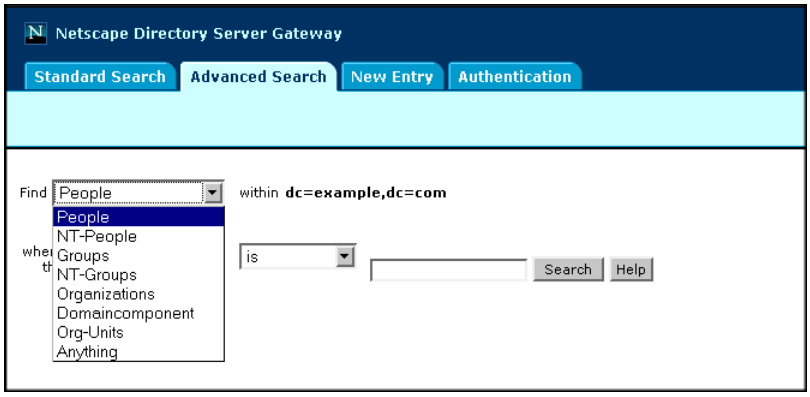


Figure 6-3 shows the attribute pop-up menu on the Advanced Search form.

Figure 6-3 Advanced Search Form: Attribute

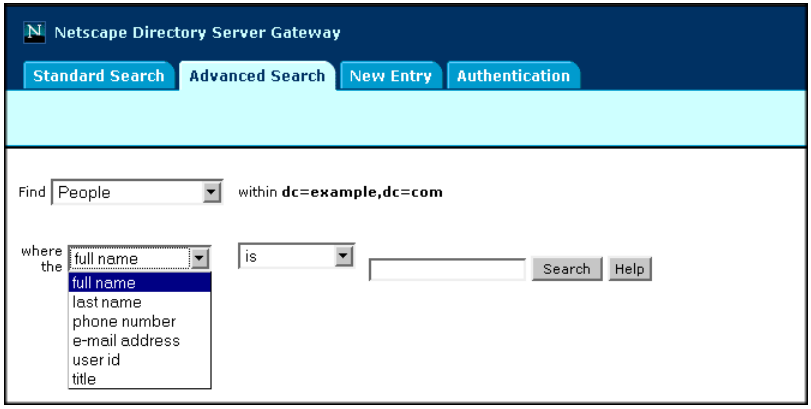
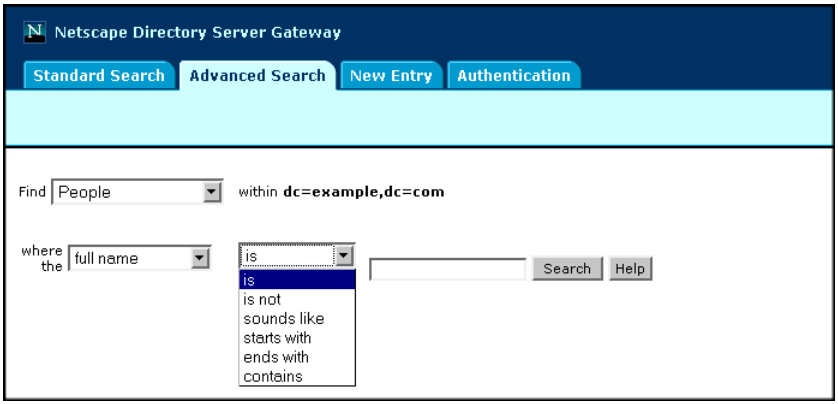


Figure 6-4 shows the matching filter pop-up menu on the Advanced Search form.

Figure 6-4 Advanced Search Form: Matching Filter



Specifying Search Attributes for Person

The `dsgwsearchprefs.conf` syntax in the following example specifies that the `cn`, `sn`, `telephoneNumber`, `mail`, `uid`, and `title` attributes will be used in a search for person entries:

```
People
""
"Search for":
"(&(objectClass=person)"
"dsgw-people"
subtree
"full name"         cn               111111  ""  ""
"last name"         sn                   111111  ""  ""
"phone number"         "telephoneNumber"     111011  ""  ""
"e-mail address"         "mail"            111111  ""  ""
"user id"         "uid"                 111111  ""  ""
"title"         title    111111 "" ""
END
"is"    "(%a=%v))"
"is not"    "(!(%a=%v)))"
"sounds like"    "(%a~=%v))"
"starts with"    "(%a=%v*))"
"ends with"    "(%a=*%v))"
"contains"    "(%a=*%v*))"
END
```

The first column in the example specifies how the LDAP attribute shown in the second column appears in the drop-down menu on the Advanced Search form.

The third column contains a string of six bits. Each bit position in the string maps to a match type, as shown in Table 6-2. A value of 1 indicates that the match type is valid for the associated attribute. A value of 0 indicates that the match type is not valid. In the example, the bit position for the telephone number attribute is set to 0, indicating that the Directory Server will not search for sounds like match types for phone number entries on the Advanced Search form.

Table 6-2 Bit Positions and Corresponding Search Match Types

Bit Position	Match Type
1	contains
2	ends with
3	starts with
4	sounds like
5	is not
6	is

The fourth and fifth columns in the search attributes contain empty strings required by the gateway. These should not be altered.

Directory Express Search Support for User ID

Directory Express does exact matches for user ID strings. It does not attempt to match user ID substrings.

To configure substring matching for user IDs, substring index the `uid` attribute, uncomment the appropriate lines in `pbconfig/dsgwfilter.conf`, and comment out the corresponding lines.

Adding Search Support for Additional Attributes

The syntax in the following example specifies preferences for searching the `pagerTelephoneNumber` attribute.

```
People
""
"Search for":
"(&(objectClass=person)
```

```
"dsgw-people"
subtree
"full name"    cn                111111  ""  ""
"last name"         sn                    111111  ""  ""
"phone number"         "telephoneNumber"      111011  ""  ""
"e-mail address"          "mail"             111111  ""  ""
"user id"         "uid"                  111111  ""  ""
"title"         title     111111  ""  ""
"pager number"          pagerTelephoneNumber     111011
""  ""
```

As a result of adding this syntax to `dsgwsearchprefs.conf`:

- A pager number selection will appear in the drop-down menu on the Advanced Search form.
- The gateway will search the `pagerTelephoneNumber` attribute of all entries within the scope of the search.
- The gateway will look for values that contain, end with, start with, or are identical to the search string entered by the user. It will not look for values that sound like the search string entered by the user.

Adding Search Support for a New Object

There are two ways to add search support for a new object:

- Update entries in `dsgwsearchprefs.conf` and `dsgwfilter.conf` with definitions of search attributes for the new object. Use this method to add search support for an object that is a child of another object.
- Create new entries in `dsgwsearchprefs.conf` and `dsgwfilter.conf` for a new object class. This method requires specifying preferences for searching object class attributes and defining a filter to use in expressing search results.

Extending Search Preferences

The syntax in the following example shows the introduction of a new object, `examplePerson`, and a new attribute, `dateOfBirth`, to the search preferences for the `person` object class.

```
People
""
"Search for":
"(&(objectClass=person)(objectClass=examplePerson)"
"dsgw-people"
subtree
"full name"         cn                111111  ""  ""
"last name"         sn                    111111  ""  ""
"phone number"         "telephoneNumber"      111011  ""  ""
"e-mail address"         "mail"             111111  ""  ""
"user id"         "uid"                  111111  ""  ""
"title"         title     111111 "" ""
"birthdate"      dateOfBirth          111011          " "     " "
```

As a result of adding the syntax shown in the example:

- The gateway will include `examplePerson` entries in its search for persons.
- A `birthdate` field will appear on the Advanced Search form.
- The gateway will search the `dateofBirth` attribute of all entries within the scope of the search.

The syntax added to `dsgwfilter.conf` in the following example defines the search values for the `dateOfBirth` attribute:

```
"[0-9][0-9]/[0-9][0-9]/[0-9]0-9]" " " "
    (dateOfBirth=%v))" "date of birth is"
    (dateOfBirth=%v*))" "date of birth starts with"
```

Creating a New Search Object

Adding a new search object that is not similar to existing search objects requires:

- A new entry in `dsgwsearchprefs.conf`.
- A new entry in the `dsgwfilter.conf`.
- A new search results template.

The syntax in the following example specifies that the search definition for `OrgRole` will search the `cn`, `l`, `roleOccupant`, `description`, and `telephoneNumber` attributes:

```
"Org-Roles"
""
"Search For:"
"(&(objectClass=organizationalRole)
"dsgw-orgrole"
```

```
not-used-by-dsgw
not-used-by-dsgw
subtree
"name"        "cn"    111111 "" ""
"location"        "l"     111111 "" ""
"occupant"        "roleOccupant"      111111 "" ""
"description"        "description"    111011 "" ""
"phone number"        "telephoneNumber"     111011 "" ""
END
"is"        "(%a=%v))"
"is not"          "(!(%a=%v)))"
"sounds like"        "(%a~=%v))"
"starts with"          "(%a=%v*))"
"ends with"         "(%a=*%v))"
"contains"         "(%a=*%v*))"
END
```

The syntax in the above example specifies a list of filters for searching attributes in the `OrgRole` object class.

This example creates a new `dsgwfilter.conf` entry for `OrgRole`:

```
dsgwfilter.conf:
dsgw-orgrole"
"="    " "    "(%v))"    "LDAP filter is"

"^[+]*[0-9][ 0-9-]*$"    "
"    "(telephoneNumber=*%v))"    "phone number ends with"
"^\*$"    "  "    "(cn=*))"    "name is"
".*"    ". _"    "(cn=%v1-))"    "name is"
        "(cn=*%v1-*))"    "name contains"
        "(cn~=%v1-))"    "name sounds like"
```

Modifying search results forms is described in “Modifying Search Result Templates,” on page 80.

Modifying Default Search Filters

The gateway uses `dsgwflter.conf` to map patterns in search strings to a relevant search filter and search result description (a search pattern is a grep-style regular expression). `dsgwwfilter.conf` can be optimized to respond to common user data patterns.

Modify existing search filters in `dsgwfilter.conf` to support new user data patterns instead of creating new filters.

The sections that follow explain:

- Search Filters for User Data Patterns
- Specifying a Search Filter for a New Object

Search Filters for User Data Patterns

This example shows typical search filter syntax for any search string containing the @ symbol. In this example, the gateway will respond to search strings containing the @ symbol (the pattern) by searching the `mail` attribute for values that are equal to or start with the supplied value (the filter). The gateway will then return a message on the search results form indicating the number of entries where the `"email address is"` or `"email address starts with"` the user-supplied value (the description).

```
"@"    " ""(mail=%v))"    "email address is"
    "(mail=%v*))"    "email address starts with"
```

NOTE Standard searches use only the filters associated with the first matching pattern. Advanced searches use all filters defined for the entry.

Specifying a Search Filter for a New Object

The syntax in the following example allows users to search person entries by birthday or birth month:

```
[0-9][0-9]/[0-9][0-9]/[0-9]0-9](dateOfBirth=%v))"date of birth
is" dateOfBirth=%v*))"date of birth starts with"
```

As a result of adding the line `dateOfBirth=%v*))date of birth starts with` to the `dsgw-people` entry in `dsgwfilter.conf`, the gateway will also filter the `dateofBirth` attribute for values that start with the supplied value (the filter). The gateway will return a message on the search results form indicating the number of entries where the `"date of birth is"` or `"date of birth starts"` `with` the user-supplied value (the description).

NOTE Make sure to place new patterns near the top of the pattern definitions for a given object. For example, in the `dsgw-people` entry, place customized patterns before the pattern that begins with the @ symbol. Patterns near the end of the entry are more general and will match many different strings.

Customizing Search Result Templates

The following sections describe how the gateway displays search results and contains procedures for customizing the gateway search result templates:

- How the Gateway Displays Search Results
- Modifying Search Result Templates

How the Gateway Displays Search Results

When a user submits a standard search or advanced search from the gateway, the gateway constructs a search string and filter for the corresponding search object and queries the Directory Server. The Directory Server responds with matching entries in the LDAP database. The gateway uses a search result template to display the entries returned by the Directory Server.

Search Result Tables

Search results are displayed as tabular data. Headings in each result table reflects the object attributes identified in the search result template.

For example, the heading row on the search results form for a People search displays the Name, User ID, Phone Number, E-Mail Address, and Group attributes.

Figure 6-5 Search Results

Netscape Directory Server Gateway
Standard Search
Advanced Search
New Entry
Authentication
Find People within dc=example,dc=com
Search for carter
Search
Help
Found 4 entries where the name or user id is 'carter'.
Name ID Phone E-mail AIM ID Group
Karen Carter kcarter +1 408 555 4675 kcarter@example.com Human Resources People
Mike Carter mcarter +1 408 555 1846 mcarter@example.com Accounting People
Sam Carter scarter +1 408 555 4798 scarter@example.com Accounting People
Stephen Carter scarte2 +1 408 555 6022 scarte2@example.com Product Development People
Click on an entry's ID to bring up more information about that entry.

Table 6-3 lists the default gateway search objects and the information displayed on the search results list. Search results templates are stored in the *serverRoot*`/clients/dsgw/config` directory and use the `list-`*search object*`.html` file naming convention.

Table 6-3 Default Search Results for Search Objects

Search Object	Search Result Template Used	Search Results Displayed
Domain Component	`list-Domaincomponent.html`	Organization name, description, and phone number.
people	`list-People.html`	Name, phone number, e-mail address, and title.
NT-people	`list-NT-People.html`	Name, Windows domain, Windows username, and phone number.
Groups	`list-Groups.html`	Group name and description.
NT-Groups	`list-NT-Groups.html`	LDAP group name, Windows domain name, Windows group name, and description.
Organizations	`list-Organizations.html`	Organization name, description, and phone number.

Table 6-3 Default Search Results for Search Objects *(Continued)*

Search Object	Search Result Template Used	Search Results Displayed
Org-Units	`list-Org-Units.html`	Organizational unit name, description, and phone number.
Anything	`list-Anything.html`	Name, phone number, e-mail address, and description.

A new search result template is required for each new object class that is not a child of another object class.

Modifying Search Result Templates

This section covers:

- Adding Information to Search Results
- Removing Information from Search Results

Adding Information to Search Results

This example shows how additional information can be added to the search result by modifying the corresponding search result template.

```
<TR>
<TH NOWRAP>Name<TH NOWRAP>Phone Number<TH NOWRAP>E-Mail
Address<TH NOWRAP>Title<TH NOWRAP>Organizational Unit
</TR>

<TR>
<TD NOWRAP>
<!-- DS_ATTRIBUTE "attr=dn" "syntax=dn" "label=" -->
onMouseOver="window.status='Click here to view this entry in
detail'; return true">
<!-- DS_ATTRIBUTE "attr=cn" "numfields=0" "defaultvalue=name"
"options=readonly" -->
</A>
<TD NOWRAP>
<!-- DS_ATTRIBUTE "attr=telephoneNumber" "syntax=tel" -->
<TD NOWRAP>
<!-- DS_ATTRIBUTE "attr=mail" "syntax=mail" -->
<TD NOWRAP>
```

```
<!-- DS_ATTRIBUTE "attr=title" --></TD>
<TD NOWRAP>
<!-- DS_ATTRIBUTE "attr=ou" "syntax=cis" --></TD>
</TR>
```

The additional HTML table heading `syntax` adds the Organizational Unit label to the heading row of the table. The additional `DS_ATTRIBUTE` directive syntax adds a cell to the body row of the table indicating that the information is stored in the `ou` attribute of the entry and the string is case insensitive.

Removing Information from Search Results

To remove information from a search result, remove the tag that creates the table head cell which labels the attribute and the tag that creates the Directory Server call for the corresponding attribute value from the corresponding `list-`*search object*`.html` file.

For example, to remove the Windows Domain attribute from the `list-NT-People.html` search results file, delete the `<TH NOWRAP>NT Domain` tag from table heading. The table cell containing the `<!-- DS_ATTRIBUTE "attr=ntuserdomainid" "syntax=ntdomain" -->` directive would also need to be removed.

Customizing Graphics and Color

This chapter describes how to change the appearance of default gateway forms. The chapter contains the following sections:

- Appearance of Gateway Forms (page 83)
- Banner Image (page 83)
- Button Images (page 84)
- Color Schemes (page 86)
- Changing Table Colors (page 88)

Appearance of Gateway Forms

The default gateway installed during Directory Server installation matches the standard appearance of Directory Server. The gateway Interface Templates can be modified to customize the appearance of the following:

- Banner Image
- Button Images
- Color Schemes

Banner Image

The default gateway banner image that appears at the top of the gateway forms is `title.gif`. This image can be replaced by a different banner image stored as `clients/dsgw/html/title.gif`.

Updating the Banner Image (title.gif)

The following sections describe how to change the dimensions of the banner image as well as how to change the banner image filename.

Changing Dimensions of Banner Image

The default banner image has a height of 40 pixels and a width of 530 pixels. When using a banner image with a different pixel height and width, change the specifications of the image in all files in `clients/dsgw/html` where the image is referenced:

- `maintitle.html`
- `authtitle.html`
- `searchtitle.html`
- `csearchtitle.html`
- `newentrytitle.html`

Changing Filename of Banner Image

Keep the default filename — `title.gif` — for the banner image. Changing the default filename of the banner image requires updating the filename in all files where the image is referenced.

NOTE Any image used to replace `title.gif` must be a true `.gif` image. Images in other formats (PICT, EPS, BPX, TIFF) will not appear as intended.

Button Images

Buttons on gateway forms can be replaced by updating button image files stored in the `clients/dsgw/html` directory. Table 7-1 describes the button image files stored in the `clients/dsgw/html` directory.

Table 7-1 Button Images

Button Image	Description
`stsearch_off.gif`	Used in the `maintitle.html`, `authtitle.html`, `csearchtitle.html`, and `newentrytitle.html` pages.

Table 7-1 Button Images *(Continued)*

Button Image	Description
`stsearch_on.gif`	Used on the `searchtitle.html` page.
`adsearch_off.gif`	Used in the `maintitle.html`, `authtitle.html`, `searchtitle.html`, and `newentrytitle.html` pages.
`adsearch_on.gif`	Used on the `csearchtitle.html` page.
`newentry_off.gif`	Used in the `maintitle.html`, `authtitle.html`, `csearchtitle.html`, and `searchtitle.html` pages.
`newentry_on.gif`	Used on the `newentrytitle.html` page.
`authen_off.gif`	Used in the `maintitle.html`, `searchtitle.html`, `csearchtitle.html`, and `newentrytitle.html` pages.
`authen_on.gif`	Used on the `authtitle.html` page.

Updating Button Images

The default button images have a height of 24 pixels and a width of 122 pixels. If the new button image uses a different pixel height and width, these specifications must be changed in all files in the `clients/dsgw/html` directory where the image is referenced:

- `maintitle.html`
- `authtitle.html`
- `searchtitle.html`
- `csearchtitle.html`
- `newentrytitle.html`

Changing the default filename of a button `.gif` file requires updating the filename in all files where the image is referenced.

NOTE Any image used to replace `button image` must be a true `.gif` image. Images in other formats (PICT, EPS, BPX, TIFF) will not appear as intended.

Color Schemes

Changing the color schemes for a form requires editing the files that make up a gateway form. The procedure for changing colors depends on the gateway file type.

- Files Controlling Colors on Gateway Forms
- Changing Colors Using BODY Tag
- Changing Colors Using Directives

Files Controlling Colors on Gateway Forms

Table 7-2 describes the gateway files that control the appearance of gateway forms. These files may need to be updated when changing the appearance of the banner image, button images, or background and body colors.

Table 7-2 Files Controlling Appearance of Gateway Forms

To Change Colors on the ...	Edit	File Type
Authentication forms	`authtitle.html`	banner
	`authPassword.html`	script
	`authSearch.html`	script
Standard search forms	`searchtitle.html`	banner
	`searchString.html`	script
	`greeting.html`	HTML
	`list-*.html`	search result
Advanced search forms	`csearchtitle.html`	banner
	`csearchAttr.html`	script
	`csearchBase.html`	script
	`csearchMatch.html`	script
	`csearchString.html`	script
	`csearchType.html`	script
	`emptyFrame.html`	HTML
	`list-*.html`	search result

Table 7-2 Files Controlling Appearance of Gateway Forms *(Continued)*

To Change Colors on the ...	Edit	File Type
New entry forms	`newentrytitle.html`	banner
	`newentryType.html`	script
	`newentryName.html`	script
	`display-*.html`	template
View or edit entry forms	`display-*.html`	template

Changing Colors Using BODY Tag

Table 7-3 lists the five standard color attributes that can be changed.

Table 7-3 Colors and Related Attributes for BODY Tag

Color	Using Attribute
Text color	TEXT
Background color	BGCOLOR
Link color	LINK
Visited link color	VLINK
Selected link color	ALINK

To adjust background colors in the banner files and other standard HTML files, edit the HTML to change the color associated with the `BGCOLOR` and `TEXT` attributes to a different RGB color value. (For example, to change background to aquamarine and text to white, use `<BODY BGCOLOR="#70DB93" TEXT="#9932CD">`).

Changing Colors Using Directives

To change background colors in script files, add a `COLORS` directive immediately before the `BODY` tag. The syntax in the following example shows the use of the `COLORS` directive to change text color to blue, background color to aqua, and the link color to red:

```
<!-- COLORS "attribute=color [attribute=color [...]]" -->
<!-- COLORS "TEXT=#0000FF BGCOLOR=#00FFFF LINK=#FF0000" >
DS_NEWENTRY_TYPE_BODY -->
```

In the example, the attribute is a standard HTML `%color` attribute, and color is an RGB color value in the form `#rrggbb` (or a standard color name, such as aquamarine).

Changing Table Colors

The following sections describe procedures for customizing the color of tables:

- Specifying Color Names and Color Codes
- Changing Color of Table Headings
- Changing the Background Color of Table Rows

Specifying Color Names and Color Codes

There are two ways to specify colors:

- Use a color value, a six digit hexadecimal number known as the red-green-blue (RGB) triplet. The RGB triplet always begins with a hash mark (#) followed by 3 2-digit codes that represent the amount of red, green, and blue that make up the color (`#rrggbb`). There are over 16 million RGB color codes.
- Use a color name. There are sixteen standard case-insensitive color names. Table 7-4 lists the sixteen standard color names and their equivalent RGB values.

Table 7-4 Sixteen Standard Colors and Their Hexadecimal Values

Color Name	Hexadecimal Value
black	#000000
silver	#C0C0C0
gray	#808080
white	#FFFFFF
maroon	#800000
red	#FF0000
purple	#80080
fuchsia	#FF00FF
green	#008000
lime	#00FF00

Table 7-4 Sixteen Standard Colors and Their Hexadecimal Values *(Continued)*

Color Name	Hexadecimal Value
olive	#808000
yellow	#FFFF00
navy	#000080
blue	#0000FF
teal	#008080
aqua	#00FFFF

Changing Color of Table Headings

To change the color of table heading text, edit the following lines within the template file:

```
<TR BGCOLOR=006666 COLSPAN=4><FONT FACE=ARIAL, HELVETICA
COLOR=WHITE>
```

so that the `BGCOLOR` value is an RGB color value in the form `#RRGGBB` or a standard color name. The font color can be changed from white to another color using the same method. Within a single template file, there may be multiple tables and consequently multiple table head rows that need to be modified to maintain a consistent color scheme.

Changing the Background Color of Table Rows

To change the color of the table body rows, edit the following line for each table within the template file:

```
<TABLE CELLSPACING="2" BORDER BGCOLOR=#F2F2F2 ...
```

so that the `BGCOLOR` attribute specifies the RGB color value in the form `#RRGGBB` or a standard color name representing the new color.

Appendix A

Parameters Defined in the .conf File

The `dsgw.conf` and `pb.conf` files are installed during Red Hat Directory Server (Directory Server) installation. This appendix describes the configuration parameters defined in these files. Associated directives are described in Appendix B, “Gateway Directives.”

authlifetime

Specifies the amount of time in seconds before a user’s authentication expires in the gateway. When authenticating to the directory from the gateway, the gateway retains authentication credentials for the amount of time specified in this parameter. Once authentication credentials have expired, the gateway prompts the user to re-authenticate.

For information on authenticating to the Directory Server using the gateway, see the online documentation that is available through the gateway.

Format

`authlifetime` *seconds*

Example

The following example causes user authentication to expire in two hours. This is the default expiration time:

```
authlifetime 7200
```

baseurl

Specifies the host name and port number used to contact the Directory Server. This parameter also determines the search base used for searches performed from the gateway and whether the gateway uses SSL to communicate with the Directory Server.

Format

`baseurl [ldap | ldaps]://`*`dirHost`*`:`*`dirPort`*`/`*`searchBase`*

ldap | ldaps. Use LDAP to have the gateway communicate the Directory Server without using SSL. Use LDAPS to have the gateway communicate with the Directory Server using SSL.

dirHost. Indicates the host name of the machine where the Directory Server is installed.

dirPort. Indicates the port number used by the Directory Server. Always specify a port number, even when using standard LDAP or LDAPS port numbers (389 and 636, respectively).

searchBase. Indicates the distinguished name (DN) representing the point in the directory from which all searches are performed. Normally, *searchBase* is set to the directory's suffix.

Substitute the following hexadecimal values for the equal sign, space, and comma in the search base:

- use %3D instead of equal sign (=)
- use %20 instead of space ( )
- use %2C instead of comma (,)

Example

The following example sets the base URL to use SSL communications to a server running on the well-known LDAP security port (`636`). The base search address is set to `o=example.com`.

```
baseurl "ldaps://dirserver.example.com:636/o%3Dexample.com"
```

binddnfile

Specifies the location of the file where the bind DN and bind password are stored. This file should be stored separately of the `.conf` file for the gateway instance. The `binddnfile` is used to authenticate to the server for non-anonymous searching.

Format

`binddnfile` *binddn_filename*

Example

```
binddnfile /export/TEST/binddnfile
```

changeHTML

Used by the gateway to substitute ideographic space for non-breaking space (nbsp) in Asian character sets.

Format

`changeHTML` *nbsp_from nbsp_to charset*

Example

```
changeHTML <space character> <space charac ter> Shift_JIS
```

charset

Defines the default character set for communication with HTTP clients. The default is UTF-8 (Unicode), which supports all the characters in the Directory Server. UTF-8 is the preferred character set; however, many browsers don't support the UTF-8 charset or display it poorly.

Some users may require a different character set than the one specified in using this parameter. For these users, the `charset` parameter setting may be overridden by creating a *LANG*`/dsgw/charset.conf` file which contains the charset name. However, to receive the correct language, users will have to configure their browsers to send the appropriate accept-language headers.

For compatibility with HTTP clients that can't handle an HTTP response with a `charset` parameter in the content-type, comment out this parameter in the `.conf` file. This prevents the gateway from sending an explicit charset to gateway clients. When no `charset` parameter is defined, the gateway by default transmits ISO-8859-1 (Latin-1).

The `charset` parameter is ignored by current versions of Netscape Communicator and Internet Explorer, which request the UTF-8 charset by default. Forcing these clients to use a non-UTF-8 charset (such as Latin-1) requires the `ignoreAcceptCharsetFrom` parameter.

More information: “ignoreAcceptCharsetFrom,” on page 97

Format

`charset` *character_set*

Example

```
charset UTF-8
```

For more information about charsets, see RFC 1345, which defines the syntax of charset names.

configdir

Specifies the location of the configuration directory of the gateway. These include the object class templates, search configuration files, search result templates, and script files used to generate HTML forms dynamically for the user.

The configuration directory for the default gateway (`dsgw.conf`) is `../config`. The configuration directory for Directory Express (`pb.conf`) is `../pbconfig`.

Format

`configdir "`*configuration_directory*`"`

Example

```
configdir "../exampleconfig"
```

dirmgr

Specifies the distinguished name of the Directory Manager. This is the DN used to bind to the Directory Server when users authenticate as the Directory Manager from the gateway. Use a DN other than the root DN for this purpose. It is intended that the DN specified here has read and write authority for the subtree that the gateway sees.

For information on authenticating as the Directory Manager from the gateway, see the online documentation that is available through the gateway.

Format

`dirmgr "`*distinguished_name*`"`

Example

```
dirmgr "cn=Directory Manager, o=example.com"
```

For information on the root DN and on setting permissions for the directory, see the *Red Hat Directory Server Administrator's Guide*.

enable-aim-presence

Specifies the AIM® presence (online or offline) of a user by displaying or hiding the AIM icon in the Directory Server Gateway UI. If `enable-aim-presence` is set to `true` and if the user being displayed is logged into the AIM service, the AIM icon show ups in the UI when the full entry for a user is being displayed.

By default, the AIM icon won't show up for lists of users because it would have adverse affects on search performance. To see AIM presence for lists of users (or multiple search results), the following files will need to be modified:

- `config/list-People.html`
- `config/list-NT-People.html`
- `pbconfig/list-People.html`

The block of text that needs to be modified is shown below:

```
<SCRIPT LANGUAGE="JavaScript">
<!-- Hide from non-JavaScript-capable browsers
showAimIcon(
<!-- DS_ATTRIBUTE "attr=nsaimid" "options=quoted" -->
,
"" //<!-- DS_ATTRIBUTE "attr=nsaimstatustext" "options=quoted"
-->
);

//// Uncomment the above DS_ATTRIBUTE directive and remove the
////
//// double quotes to have aim presence in search results lists
////
```

The modified block of text is shown below (the `""` and `//` removed from the sixth line):

```
<SCRIPT LANGUAGE="JavaScript">
<!-- Hide from non-JavaScript-capable browsers
showAimIcon(
<!-- DS_ATTRIBUTE "attr=nsaimid" "options=quoted" -->
```

```
,
<!-- DS_ATTRIBUTE "attr=nsaimstatustext" "options=quoted" -->
);

//// Uncomment the above DS_ATTRIBUTE directive and remove the
////
//// double quotes to have aim presence in search results lists
////
```

Once this is done, listings of multiple users will show AIM presence for each user.

NOTE By default, `nsaimid` and `nsaimstatustext` are used for AIM ID and AIM presence information, respectively. If you use different attributes, be sure to change the HTML files.

Format

```
enable-aim-presence true | false
```

Example

```
enable-aim-presence true
```

gwnametrans

Used by the gateway CGI scripts to specify the URL to output for HTTP redirection. This needs to be specified as "`/clients/dsgw/`*htmldir*" and should be the same as the `NameTrans` set in the HTTP server, if any is being used.

Format

`gwenametrans "`*HTTP_redirect*"

Example

```
gwnametrans "/clients/dsgw/pbhtml/"
```

htmldir

Specifies the location of the HTML files for the gateway. These include the HTML files controlling the appearance of gateway forms.

The HTML directory for the default gateway (`dsgw.conf`) is `../html`. The HTML directory for Directory Express (`pb.conf`) is `../pbhtml`.

Format

`htmldir "`*`html_directory`*`"`

Example

```
htmldir "/exampleconfig"
```

ignoreAcceptCharsetFrom

Ignores request headers for the UTF-8 character set automatically sent by Netscape Communicator and Internet Explorer browsers. Can be used together with the `charset` parameter to transmit a charset other than Unicode to all gateway clients.

Format

`ignoreAcceptCharsetFrom` *`HTTP_ client_version_string`*

Example

```
ignoreAcceptCharsetFrom Mozilla/4.01x-NSCP Mozilla/3
```

More information: “charset,” on page 93.

include

Specifies the location of another configuration file that should be read by the gateway.

Format

`include "`*`configuration_file`*`"`

Example

```
include "../config/dsgw-l10n.conf"
```

location

Defines the location choices selectable from the gateway when adding new entries. Each `location` parameter represents a branch point in the directory tree below which new entries can be added.

Format

`location` *handle* "*friendly_name*""*dn*"

handle. An arbitrary string used by the `location-suffix` parameter to map a type of entry to the locations where the entry can be created. For more information, see “location-suffix,” on page 99.

friendly_name. An arbitrary string that represents the location. This string should describe the location because the gateway displays this string to users to represent the location.

dn. The distinguished name (DN) representing this branch point in the directory. If this value is not terminated with a pound sign, the value specified on the `include` parameter is appended to this value to build the fully qualified distinguished name. If the DN is terminated with a pound sign (#), the value represented here is assumed to be a fully qualified distinguished name, and the pound sign is stripped from the distinguished name before the DN is used by the gateway.

For more information, see “include,” on page 98.

Example

The following example defines an entry creation location in a user directory. This location corresponds to the Marketing organizational unit, and the remainder of the distinguished name is built from the value set in the `include` parameter:

```
location marketing "Marketing Organization" "ou=Marketing"
```

A slightly different example defines the same location but specifies the fully qualified distinguished name:

```
location marketing "Marketing Organization" "ou=Marketing,
o=example.com#"
```

For a more complete example of the `location` parameter, see “Mapping Locations and Entry Types,” on page 57.

location-suffix

Identifies the directory suffix used to create new entries in the directory.

This value is appended to the DN field of the `NLS` parameter when the gateway is used to create new entries in the directory.

Format

`location-suffix "`*`suffix`*`"`

Example

```
location-suffix "o=example.com"
```

newtype

Defines the types of entries that can be added to the directory using the gateway. `newtype` also defines the locations in the directory where an entry type can be added. For a user to create the entry, the corresponding location must be defined using the `location` parameter.

Format

`newtype` *template_name* `"`*friendly_name*`"` *rdnattr locations*

template_name. The name of a `display-`*template_name*`.html` file that defines the object class listed. Template files are stored in the `../config` directory. The gateway uses these files to define how various types of entries are displayed when entries are being created or viewed:

- orgperson — Corresponds to the `display-orgperson.html` template. Defines how the gateway displays an entry of object class type `inetOrgPerson`.
- groupun — Corresponds to the `display-groupun.html` template. Defines how the gateway displays an entry of object class type `groupOfUniqueNames`.
- orgunit — Corresponds to the `display-orgunit.html template`. Defines how the gateway displays an entry of object class type `organizationalUnit`.
- org — Corresponds to the `display-org.html` template. Defines how the gateway displays an entry of object class type `organization`.

friendly_name. An arbitrary string that describes the entry. This string should be reasonably descriptive of the entry type because the gateway displays this string to users who are adding entries.

rdnattr. The attribute used to name entries of this type. For example, the default value for the `rdnattr` field for people entries is `uid`. This means that any people entries created using the gateway will have DNs of the following format:

`uid=`*`string`*

The `rdnattr` field can be modified so that entries are named using a different attribute. For example, to change the `rdnattr` of the `newtype orgperson` line from `uid` to `cn`, people entries created using the gateway will have `cn`-based DNs rather than the UID-based DNs (the default setting).

locations. A space-separated list of the locations where this type of entry can be added. The locations in this list must be identical to the *handle* specified on the corresponding `location` parameter.

Example

The following example allows persons to be added to the Marketing subtree using the template for `organizationalPerson`:

```
newtype orgperson"Person"cn marketing
```

For a more complete example of the `newtype` parameter, see “Mapping Locations and Entry Types,” on page 57.

NLS

Identifies the libNLS data directory, which should contain a directory named “locales,” containing the configuration files `LANG.ctx`, `LANG.col`, and `LANG.txt` for each supported language (locale).

Format

`NLS` *libNLS_data_directory*

Example

```
NLS ../../lib/nls
```

orgchart-attrib-farleft-rdn

Specifies the attribute to be used as the leftmost RDN for the DNs of user entries (in order to enable the Org Chart application to search for entries).

The `orgchart-attrib-farleft-rdn` attribute is the same as the one included in the Org Chart's configuration file (*serverRoot*`/clients/orgchart/config.txt`), and the attribute value (`uid`, `cn`, and so on) must match the values specified in the `config.txt` file.

Format

`orgchart-attrib-farleft-rdn` *attribute*

Example

```
orgchart-attrib-farleft-rdn uid
```

securitypath

Identifies the location of the certificate database used by the gateway when using SSL to communicate with the Directory Server. The certificate database contains the Certificate Authority issuing the certificate for the Directory Server.

Format

`securitypath "/usr/redhat/servers/alias/`*cert*`.db"`

Example

```
securitypath "/usr/redhat/servers/alias/pb-cert.db"
```

template

Maps specific object classes to internal gateway templates. These templates define how a specific object class such as a person, a group, or an organizational unit is displayed in the gateway. The `templatename` identified has a corresponding HTML template stored in `clients/dsgw/config/`.

Format

`template` *template_name object_class*

Example

The following example identifies `orgperson` as the template defining attributes for the `person` and `inetorgperson` object classes:

```
template orgperson person inetorgperson
```

url-orgchart-base

Points to the Org Chart application, providing a link to the Org Chart application from the Directory Server Gateway UI pages. By default, the Directory Server installation program sets the base to use the Red Hat Administration Server as the web server. You can change the host name and port number to be that of a dedicated web server. (See “HTTP Server Configuration,” on page 32.)

In the absence of the `url-orgchart-base` configuration field, there will be no link to the Org Chart application in the Directory Server Gateway UI.

The Org Chart application also has a similar URL which points to the Directory Server Gateway (the `dsgw` instance). You can change it to `..../lang?context=pb` to have the Org Chart link to the Phonebook or remove the `context=dsgw` altogether and have it link to the default gateway instance (`default.conf`, which is the `dsgw` instance).

Format

`url-orgchart-base`
`http://`*`adminHost`*`:`*`adminPort`*`/clients/orgchart/bin/org?context=dsgw&data=`

adminHost. Specifies the host name of the machine on which the Red Hat Administration Server is installed.

adminPort. Specifies the port number used by the Red Hat Administration Server.

Example

```
http://machine.example.com:24020/clients/orgchart/bin/org?context=
dsgw&data=
```

vcard-property

Directory Server Gateway allows users to view vCards for `person` and Windows person directory entries. The vCard and LDAP specifications define different labels to access information: vCards use properties, and LDAP uses attributes. Therefore, there must be a way to map the vCard property names to the LDAP attribute names so that the Directory Server can locate the information for the vCard display. The `vcard-property` parameter accomplishes mapping vCard property to LDAP attribute.

Format

`vcard-property` *vcardprop syntax ldapattr* [*ldapattr2...*]

vcardprop. The name of a vCard property. vCard properties that are currently mapped to LDAP attributes are:

- FN — The Formatted Name property. All vCards must have an FN property. By default, FN is mapped to the `cn` attribute.
- N — The Name property. By default, N is mapped to the `sn` and `givenName` attributes.
- ORG — The ORG property may refer to the organizational name and units of the person or resource associated with the vCard. By default, ORG is mapped to the `o` and `ou` attributes.
- ROLE — The ROLE property may refer to the role, occupation or business category of the person or resource described by the vCard. By default, ROLE is mapped to the `businessCategory` attribute.
- ADR;WORK — The work address of the of the person or resource described by the vCard. By default, ADR;WORK is mapped to the `postalAddress` attribute.
- ADR;HOME — The home address of the of the person or resource described by the vCard. By default, ADR;HOME is mapped to the `homePostalAddress` attribute.
- EMAIL;INTERNET — The email address of the person or resource described by the vCard. By default, EMAIL;INTERNET is mapped to the `mail` attribute.
- TITLE — The TITLE property specifies the job title, functional position or function of the person or resource described by the vCard. By default, TITLE is mapped to the `title` attribute.
- TEL;WORK — The business telephone number of the person or resource described by the vCard. By default, TEL;WORK is mapped to the `telephoneNumber` attribute.
- TEL;FAX — The fax number of the person or resource described by the vCard. By default, TEL;FAX is mapped to the `facsimileTelephoneNumber` attribute.
- TEL;CELL — The cellular telephone number of the person or resource described by the vCard. By default, TEL;CELL is mapped to the `mobile` attribute.
- TEL;HOME — The residential telephone number of the person or resource described by the vCard. By default, TEL;HOME is mapped to the `homePhone` attribute.
- NOTE — Provides any additional comments or information about the person or resource described by the vCard. By default, NOTE is mapped to the `description` attribute.

syntax. A string that describes the nature of the vCard information. The following syntaxes are supported:

- cis — used for simple strings, such as a person’s name or telephone number.
- mls — used for multi-line strings, such as a mailing address.

ldapattr [ldapattr2...]. The attribute(s) to be mapped to the vCard property. This is useful when mapping a vCard property to a custom attribute.

Example

The following example changes the mapping of the NOTE property from the default `description` attribute to a custom attribute named `hobbies`:

```
vcard-property NOTE mls hobbies
```

Appendix B

Gateway Directives

This appendix describes directives used in gateway HTML object class and search result templates. The appendix contains the following sections:

- Introduction (page 107)
- Context-Related Directives (page 109)
- Entry-Related Directives (page 110)
- Miscellaneous Directives (page 122)

Introduction

The display of LDAP directory information is controlled by HTML template files containing directives. Directives are HTML comments that can be interpreted by the gateway CGIs.

The most commonly used directive is `DS_ATTRIBUTE`, used to display attributes present in LDAP entries. Here are some other examples of directives:

```
<!-- DS_HELPBUTTON "topic=HELP-ME-NOW" -->
<!-- DS_ATTRIBUTE "attr=sn" "size=>20" -->
<!-- IF "BoundAsThisEntry" -->
<!-- ENDIF -->
```

NOTE With the exception of `GCONTEXT` and `PCONTEXT`, each directive must start at the beginning of a line and be contained on a single line in the HTML file. Most of the Directory Server Gateway directives begin with `DS_`.

Structure of an HTML Template

Directory entry display, edit, and add templates generally have the following structure:

```
<HTML>
<HEAD>
<!-- DS_ENTRYBEGIN -->
<!-- DS_EMIT_BASE_HREF -->
<!-- BODY -->
<!-- DS_LAST_OP_INFO -->
<!-- DS_BEGIN_ENTRYFORM -->
<!-- attribute directives, e.g., -->
<!-- DS_ATTRIBUTE "attr=givenName" "size=>20" -->
<!-- DS_ATTRIBUTE "attr=sn" "size=>20" -->
<!-- etc. -->
<!-- DS_SAVEBUTTON "label= SAVE " -->
<!-- DS_END_ENTRYFORM -->
<!-- DS_ENTRYEND -->
<!-- ENDHTML -->
```

Structure of an HTML Template for Directory List

Directory entry list templates generally have the following structure:

```
<HTML>
<!-- TITLE "Search Results" -->
<!-- DS_SEARCHDESC -->
<!-- IF "FoundEntries" -->
<!-- DS_SORTENTRIES "attr=XXX" -->
<!-- DS_ENTRY_BEGIN -->
<!-- stuff that is repeated for each entry found, e.g., -->
<!-- DS_ATTRIBUTE "attr=dn" "syntax=dn" -->
<!-- etc. -->
<!-- DS_ENTRYEND -->
<!-- ELSE -->
<!-- stuff to be rendered if no entries were found, e.g.,-->
Please try a different search....
<!-- ENDIF -->
<!-- ENDHTML -->
```

Context-Related Directives

The context-related directives `GCONTEXT` and `PCONTEXT` appear within a line and are not required to appear at the beginning of a line. This is an exception to the rule. All other directives must appear at the beginning of a line to be recognized by the Directory Server.

GCONTEXT

The `<!-- GCONTEXT-->` directive appears within a URL and is used in the invocation of CGIs through GET operations. `<!-- GCONTEXT-->` can appear anywhere on a line, and more than once within a line. The gateway CGI reading `<!--GCONTEXT -->` replaces it with the gateway context it has at the time.

Arguments

None.

Example

```
<a href=/clients/dsgw/bin/lang?<?-- GCONTEXT
-->&file=auth.html>click</a>
```

PCONTEXT

The `<!-- PCONTEXT-->` directive must appear on a line by itself. The gateway CGI reading `<!--PCONTEXT -->` replaces it with a hidden variable indicating the context it has at the time.

Arguments

None.

Example

```
<form method=post action=/dsgw.bin/dosearch>
<input type=hidden name=dn valute=””>
<!-- PCONTEXT -->
<form>
```

Entry-Related Directives

Entry-related directives are supported by the `dosearch` and edit CGIs.

DS_ENTRYBEGIN

Delimits the beginning of an entry. The `DS_ENTRYBEGIN` directive is used in display or edit templates to mark the start of an LDAP entry and in list templates to mark the beginning of a section which should be repeated for each entry which is returned by the search. Always paired with `DS_ENTRYEND`.

Arguments

None.

DS_ENTRYEND

Delimits the end of an entry. Always paired with `DS_ENTRYBEGIN`.

Arguments

None.

DS_ATTRIBUTE

The `DS_ATTRIBUTE` directive is replaced with the contents of an attribute (its values). This directive must appear within a `DS_ENTRYBEGIN...DS_ENTRYEND` block.

Arguments

attr=attribute-name. Displays the named attribute. Any attribute may be displayed. The special attribute `dn` is recognized and causes the distinguished name of the entry to be displayed.

syntax=syntax-type. Displays the attribute as if it were of `syntax=syntax-type`. If no `syntax=` argument is given, it is assumed to be `syntax=cis`. Legal values are described in Table B-1.

Table B-1 DS_ATTRIBUTE: Display of syntax Argument

syntax	Description	Display As
`tel`	Display as a telephone number	text
`dn`	Display as a distinguished name	href (a link to an LDAP entry)
`mail`	Display as a `mailto:` URL	href (`mailto:` *URL*)
`mls`	Display as a multi-line string	text
`time`	Display as date/time	text
`cis`	Display as a case-ignore string	text
`url`	Display as a labeled URL	href (URL)

type=how-to-display. Renders the attribute on-screen in a particular format. Legal values described in Table B-2 correspond roughly to HTML form element names.

Table B-2 DS_ATTRIBUTE: Display of type Argument

type	Display
`text`	Display as text.
`textarea`	Show as an HTML TEXTAREA.
`radio`	Show as a radio button.
`checkbox`	Show as a check box.
`password`	Show as an HTML password text box (characters are not echoed).
`hidden`	Show values in hidden form fields.

options=option. Modifies how the attribute is displayed. Legal values are described in Table B-3.

Table B-3 DS_ATTRIBUTE: Display of options Argument

options	Display
`sort`	Sort the attribute values.
`nolink`	Do not attempt to display the attribute as a hyperlink.

Table B-3 DS_ATTRIBUTE: Display of options Argument *(Continued)*

options	Display
`dntags`	Applies only when using `syntax=dn` — tags are displayed when showing DNs. Normally, they are not displayed.
`dateonly`	Applies only when using `syntax=time` — only displays the date, omitting the time.
`readonly`	When editing, do not allow the user to modify the attribute's value.
`dnpicker`	Applies only when using `syntax=dn` — embeds delete checkboxes and Javascript array information. Needed for "Find and Add."
`unique`	Enforce uniqueness when adding or editing values.
`quoted`	Applies when using Javascript — to have the value returned be surrounded by quotes.

defaultvalue=default-value. Supplies a default value for the attribute, which is shown if no attribute was read from the Directory Server.

within=string-to-embed-in. For each value, outputs the text in `string-to-embed-in`, replacing all occurrences of the `string --value--` with an attribute value.

href=href. Specifies the HREF used for the hyperlink. For example, you can specify anonMouseOver JavaScript handler using the `href=` option.

hrefextra=extra-text. Specifies additional text which is inserted after the closing quote of the HREF tag.

dncomponents=number. Gives the number of DN components to show when displaying a DN. For example, if you include `dncomponents=2` and display the DN `cn=James Doe,o=Example Corporation,c=US`, the output will be `James Doe, Example Corporation`.

size=number. Same as `cols` argument.

rows=number, rows=+number, rows=>number. Controls the number of rows used to display the entry. For `type=text`, this controls the number of editable HTML INPUT fields. For `type=textarea`, this controls the number of rows in the text area. If number is preceded by a plus (+) sign, then number extra rows are included. If the number is preceded by a greater-than sign, then at least that number of rows is included.

cols=number, cols=+number, cols=>number. Controls the width of the displayed attribute. If a number is given by itself, then the attribute is displayed with exactly that number of columns. If a plus (+) sign is given before the number, then the attribute is given that number number of extra columns. For example, if the value is 10 characters wide, and the number is 10, then 20 columns are used when displaying the number. If a greater-than sign (>) is given before number, then the displayed width is at least that number of columns.

numfields=number, numfields=+number, numfields=>number. Controls the number of editable fields displayed when editing. If the number is preceded by a plus (+) sign, then the number of fields displayed is however many values were read from the server plus number. If the number is preceded by a greater-than sign (>), then at least that numberof values are displayed when editing.

true=string. Label used for Boolean values that are true.

false=string. Label used for Boolean values that are false.

value=string. Value associated with an instance of a checkbox that is used to display strings values (not syntax=bool values).

Examples

```
<!-- DS_ATTRIBUTE "attr=dn" "syntax=dn" "dncomponents=2"
"options=nolink" -->

<!-- DS_ATTRIBUTE "attr=givenName" "cols=>32" -->

<!-- DS_ATTRIBUTE "attr=sn" "cols=>32" -->

<!-- DS_ATTRIBUTE "attr=uid" "numfields=1" "cols=>16"
"options=unique" -->

<!-- DS_ATTRIBUTE "attr=mail" "syntax=mail" "cols=>20" -->

<!-- DS_ATTRIBUTE "attr=telephoneNumber" "syntax=tel" "cols=>16"
"numfields=+1" -->

<!-- DS_ATTRIBUTE "attr=modifyTimestamp" "syntax=time"
"defaultvalue=N/A" "options=readonly" -->

<!-- DS_ATTRIBUTE "attr=modifiersName" "syntax=dn"
"defaultvalue=N/A" "options=readonly" -->

<!-- DS_ATTRIBUTE "attr=mailDeliveryOption" "type=CHECKBOX"
"value=mailbox" -->

<!-- DS_ATTRIBUTE "attr=mailDeliveryOption" "type=CHECKBOX"
"value=native" -->

<!-- DS_ATTRIBUTE "attr=mailForwardingAddress" "syntax=mail"
"type=textarea" "rows=2" "cols=30" -->
```

DS_OBJECTCLASS

Describes the type of directory entries for which a given template should be used.

Arguments

value=value1,value2,...valueN. Specifies a list of object class values. For a template file to be used to display a given entry, all of the values given must be values in the entry's `objectClass` attribute.

NOTE The gateway does not read the template files to determine which template to use. Instead, it reads the `dsgw.conf` file and scans the template lines in that file.

Example

```
<!-- DS_OBJECTCLASS "value=person,inetOrgPerson" -->
```

DS_VIEW_SWITCHER

Display a widget that provides access to all views that are appropriate for this entry. Usually this directive will be used without any arguments at all, which causes a table that contains one cell for each available view to be displayed.

Arguments

prefix=text. HTML text to emit before view elements (optional).

suffix=text. HTML text to emit after view elements (optional).

curprefix=text. HTML text to emit before the link to the current (active) view element (optional).

cursuffix=text. HTML text to emit after the link to the current view element (optional).

altprefix=text. HTML text to emit before each link to an alternative view element (optional).

altsuffix=text. HTML text to emit after each link to an alternative view element (optional).

Example

```
<!-- DS_VIEW_SWITCHER -->
```

DS_SORTENTRIES

Specifies that entries should be sorted; typically used within list templates. This directive must appear within a `DS_ENTRYBEGIN...DS_ENTRYEND` block. Up to two `DS_SORTENTRIES` directives are honored (the attribute from the first one that appears is used as the primary sort key, and the second one is used as a secondary sort key).

Arguments

attr=attrname. Sort the entries in ascending order by *attrname*.

Example

To sort a list of entries by common name:

```
<!-- DS_SORTENTRIES "attr=cn" -->
```

DS_SEARCHDESC

Specifies that text describing the type of search done should be displayed. For example, “Found 14 entries where the phone number ends with '25.'”

Arguments

None.

DS_POSTEDVALUE

Echoes the contents of an arbitrary posted form variable within a `VALUE=` parameter.

Arguments

name=varname. The name of the form variable.

Example

If a variable called `searchstring` is posted and contains the text John Doe, the directive

```
<!-- DS_POSTED_VALUE "name=searchstring" -->
```

will produce the following

```
HTML: VALUE="John Doe"
```

DS_EDITBUTTON

Displays a button which, when clicked, brings up an editable view of an entry. This directive must appear within a `DS_ENTRYBEGIN...DS_ENTRYEND` block. Typically used in display templates.

Arguments

label=text. Use *text* as the label on the button. If not provided, the text `Edit` is used.

Example

```
<!-- DS_EDITBUTTON "label=Edit Person" -->
```

DS_DELETEBUTTON

Displays a button which, when clicked, allows deletion of an entry. This directive must appear within a `DS_ENTRYBEGIN...DS_ENTRYEND` block. Typically used in edit templates.

Arguments

label=text. Use *text* as the label on the button. If not provided, the text `Delete` is used.

Example

```
<!-- DS_DELETEBUTTON "label=Remove Person" -->
```

DS_SAVEBUTTON

Displays a button which, when clicked, saves changes to an entry. Typically used in edit templates. This directive must appear within a `DS_ENTRYBEGIN... DS_ENTRYEND` block.

Arguments

label=text. Use *text* as the label on the button. If not provided, the text `Save` is used.

checksubmit=javascript. Submit changes only if the javascript expression is `true`.

Examples

```
<!-- DS_SAVEBUTTON "label=Save Changes" -->
```

```
<!-- DS_SAVEBUTTON "checksubmit=formDataValid()" -->
```

DS_EDITASBUTTON

Displays a button which, when clicked, allows editing of an entry using a non-default template. This directive must appear within a `DS_ENTRYBEGIN... DS_ENTRYEND` block.

Arguments

label=text. Use *text* as the label on the button. If not provided, the text `Edit As` is used.

template=template-name. Use the template name `template-name` when editing.

Example

A button to bring up `edit-passwd.html` template:

```
<!-- DS_EDITASBUTTON "label=Change Password" "template=passwd" -->
```

DS_NEWPASSWORD

Displays an HTML password INPUT field. This directive must appear within a `DS_ENTRYBEGIN...DS_ENTRYEND` block.

Arguments

None.

DS_CONFIRM_NEWPASSWORD

Displays an HTML password INPUT field. The gateway compares the value supplied by the user in this field to the value in the `DS_NEWPASSWORD` field and saves only the new password value if the two match. This directive must appear within a `DS_ENTRYBEGIN...DS_ENTRYEND` block.

Arguments

None.

DS_OLDPASSWORD

Displays an HTML password field for the old password. This directive must appear within a `DS_ENTRYBEGIN...DS_ENTRYEND` block.

Arguments

None.

DS_HELPBUTTON

Displays a help button.

Arguments

topic=topic_name. Causes the Help System to open the given topic name.

Example

```
<!-- DS_HELPBUTTON "topic=MODIFYPASSWD" -->
```

DS_CLOSEBUTTON

Displays a Close button, which causes the containing window to be closed.

Arguments

label=text. Use *text* as the label on the button. If not provided, the text `Close Window` is used.

Example

```
<!-- DS_CLOSEBUTTON "label=Cancel" -->
```

DS_BEGIN_ENTRYFORM

Causes the gateway to emit an `HTML FORM` directive and several hidden form elements which are required for proper operation of the gateway. This directive must appear within a `DS_ENTRYBEGIN...DS_ENTRYEND` block.

Arguments

None.

DS_END_ENTRYFORM

Causes the gateway to emit a `</FORM>` tag. This directive must appear within a `DS_ENTRYBEGIN...DS_ENTRYEND` block.

Arguments

None.

DS_EMIT_BASE_HREF

Emit a `<BASE>` tag that contains the base URL for the CGI that was executed.

Arguments

None.

DS_DNEDITBUTTON

Used to edit DN-valued attributes, such as group member.

Arguments

label=

template=

attr=

desc=

DS_BEGIN_DNSEARCHFORM

Used to edit DN-valued attributes, such as group member.

Arguments

None.

DS_ATTRVAL_SET

Display an attribute based on an `attrvset` as defined in the `dsgw.conf` file.

Arguments

set=name. Use information from attribute valueset name.

prefix=text. HTML text to emit before each attribute value element (optional).

suffix=text. HTML text to emit after each attribute value element (optional).

Plus any of the arguments supported by the `DS_ATTRIBUTE` directive.

Example

```
<!-- DS_ATTRVAL_SET "set=CAL" "attr=nsLicensedFor" "type=checkbox"
"prefix=<TR><TD>" "suffix=</TD></TR>" -->
```

IF/ ELSE/ ELIF/ ENDIF

Set of directives that can be used to include HTML text conditionally.

Arguments for IF and ELIF

condition. Boolean condition; if true, include following block of text.

!condition. Boolean condition; if false, include following block of text.

Arguments for ELSE and ENDIF

None.

Table B-4 Conditions Supported for ELSE and ENDIF

Condition	Arguments	Description
`FoundEntries`	none	Are there any entries being displayed?
`Adding`	none	Is the entry being edited a new entry?
`Editing`	none	Are we editing an entry?

Table B-4 Conditions Supported for ELSE and ENDIF *(Continued)*

Condition	Arguments	Description
`Displaying`	none	Are we just displaying an entry?
`Bound`	none	Is the user authenticated?
`BoundAsThisEntry`	none	Is the user authenticated as the entry we are displaying?
`AttributeHasValues`	`attr mincount`	Does the attribute `attr` have at least `mincount` values?
`AttributeHasThisValue`	`attr syntax value`	Does the attribute `attr` with syntax `syntax` have `value` as one of its values?
`AdminServer *`	none	Are we running under the Administration Server?
`DirectoryIsLocalDB *`	none	Is the Directory Server using the LDAP local database?
`PostedFormValue *`	`name value`	Is a form variable called `name` present that has `value` as its value?

NOTE Conditions marked with an asterisks (*) are supported in all the directory gateway CGIs, not just `dosearch` and `edit`.

Examples

```
<!-- IF "!DirectoryIsLocalDB" -->
The entry was last modified by <!-- DS_ATTRIBUTE
"attr=modifiersName" "syntax=dn" "defaultvalue=N/A"
"options=readonly" -->
<!-- ENDIF ---->

<!-- IF "AttributeHasThisValue" "objectclass" "cis" "mailRecipient"
--> // this entry is a mail recipient... do something special here
<!-- ENDIF ---->
```

Miscellaneous Directives

BODY

Emit HTML <BODY> element that includes color information.

Arguments

extrahtml

Examples

```
<!-- BODY  -->
<!-- BODY "onLoad=setDefaults()" -->
```

COLORS

Set color information to be used in subsequent BODY directives.

Arguments

html-color-info

Example

```
<!-- COLORS "TEXT=#000000 BGCOLOR=#FFFFFF LINK=#FF0000
VLINK=#8000FF ALINK=#FF0000" -->
```

TITLE

Emit HTML <HEAD>, <TITLE>, and <BODY> elements.

Arguments

title-string

Example

```
<!-- TITLE "Search Results" -->
```

ENDHTML

Emit </BODY></HTML> sequence

Arguments

None.

HELPBUTTON

Display a Help button (same effect as `DS_HELPBUTTON` directive but can be used from any gateway directory CGI).

Arguments

topic

Example

```
<!-- HELPBUTTON "MODIFYPASSWD" -->
```

INCLUDE

Include the contents of another HTML file. You cannot nest `include` directives.

Arguments

filename. The name of the file to include. This is relative to the `html/` directory where files such as `display-inetorgperson.html` are located.

Example

```
<!-- INCLUDE extra.html -->
```

INCLUDECONFIG

Include the contents of an HTML-based configuration file. You cannot nest `include` directives.

Arguments

filename. The name of the file to include. This is relative to the `config/` directory where files such as `dsgw.conf` are located.

Example

```
<!-- INCLUDE dsgw-orgperson.conf -->
```

DS_LAST_OP_INFO

Display a string that shows the result of the last `domodify` run. This directive works only when the `genscreen` or `edit` CGIs are invoked via `domodify`'s `completion_javascript` feature.

Arguments

prefix=prefix-text. Text displayed before the last operation info.

suffix=suffix-text. Text displayed after the last operation info.

Example

```
<!-- DS_LAST_OP_INFO "prefix=<P><FONT SIZE=%2B1>The user "
"suffix=</FONT>" -->
```

DS_LOCATIONPOPUP

Emit an HTML form element that contains a list of all the `o`'s and `ou`'s that are in the directory. If there is only one, a hidden field is produced; otherwise, an HTML select field is produced.

Arguments

name=varname. The name of the form element that is emitted.

prefix=select_prefix. Text that is output before a select element.

suffix=select_prefix. Text output after a select element.

Example

```
<!-- DS_LOCATIONPOPUP "name=base" "prefix=Choose a searchbase" -->
```

DS_GATEWAY_VERSION

Emit a string containing the version of the directory gateway CGI being executed.

Arguments

None.

Example

```
<!-- DS_GATEWAY_VERSION -->
```

IF/ ELSE/ ELIF/ ENDIF

Same as those supported by the `dosearch` and `edit` CGIs. However, conditionals marked with an asterisk (*) are supported.

Appendix C

CGI Usage

This appendix provides regular expression-type descriptions of the ways to invoke the CGIs in script files. Each regular expression is followed by the variables the expression can take on a POST. Upper case words are variables. Lower case words are literals.

auth

`auth[?context=CONTEXT[&dn=DN]]`

`authasrootdn = "true"`

or

`escapedbinddn = DN`

lang

`lang?context=CONTEXT[&file=FILE]`

search

`search?context=CONTEXT[&file=string]`

(GETs only. No POSTs.)

csearch

`csearch[?context=CONTEXT[&file=FILE]]` where `FILE` can be `type|attr|match|string|base`

```
searchType = SEARCHTYPE

searchAttr = SEARCHATTR
```

unauth

```
unauth[?conetxt=CONTEXT]
```

dnedit

```
dnedit?CONTEXT=context&TEMPLATE=tmplname&DN=dn&ATTR=attrname&DESC=
description
```

edit

```
edit?template&context=CONTEXT[&info=INFOSTRING]
[&ADD][&dn=DN][&dnattr=ATTRNAME&dndesc=DESCRIPTION]
```

(GETs only. No POSTs.)

doauth

```
escapedbinddn = DN

authdesturl = AUTHDESTURL

password = PASSWORD
```

domodify

```
changetype = CHANGETYPE

dn = DN

newrdn = RDN

completion_javascript = COMPL_JS

newpasswd = NEW_PASSWD

passwd = PASSWD

newpasswdconfirm = NPCONFIRM
```

newentry

```
newentry[?context=CONTEXT[&file=FILE]]
entrytype = ET
entryname = EN
rdntag = RDNTAG
selectLocation = SL
dnsuffix = DNSUFFIX
```

dosearch

```
dosearch?context=BLAH[&hp=host[:port]][&dn=baseDN][&ldq=LDAPquery]]
mode = MODE
searchstring = SEARCH
STRING
type = TYPE
base = BASE
attr = ATTR
match = MATCH
attr = ATTR
filterpattern = FILTERPATTERN
filterprefix = FILTERPREFIX
filtersuffix = FILTERSUFFIX
scope = SCOPE
searchdesc = SEARCHDESC
```

Index

A

Advanced search
  attributes for 69
  example of pop-up menu 71
Attributes
  adding to object classes 61
  deleting from object classes 61
auth CGI 127
Authentication
  and non-anonymous searching 27
  as directory manager 39
authlifetime parameter 91

B

Banner files
  controlling appearance of gateway forms 54
Banner image 84
baseurl parameter 92
Bind DN 27, 31
Bind password 27, 31
binddnfile
  security for 27
binddnfile parameter 93
Button images
  on gateway forms 84
  updating 85

C

CGIs
  auth 127
  csearch 127
  dnedit 128
  dosearch 129
  edit 128
  lang 127
  newentry 129
  search 127
  specifying gateway configuration to 26
  unauth 128
  usage 127
changeHTML parameter 93
Chapter Single Template 91, 107, 127
charset parameter 93
configdir parameter 94
csearch CGI 127

D

Default gateway
  .conf file for 22
  dsgw.conf 51
  forms on 25
Directives
  BODY 122
  COLORS 122
  context related 109
  DS_ATTRIBUTE 110

DS_ATTRVAL_SET 120
DS_BEGIN_DNSEARCHFORM 120
DS_BEGIN_ENTRYFORM 119
DS_CLOSEBUTTON 118
DS_CONFIRM_NEWPASSWORD 118
DS_DELETEBUTTON 116
DS_DNEDITBUTTON 119
DS_EDITASBUTTON 117
DS_EDITBUTTON 116
DS_EMIT_BASE_HREF 119
DS_END_ENTRYFORM 119
DS_ENTRYBEGIN 110
DS_ENTRYEND 110
DS_GATEWAY_VERSION 125
DS_HELPBUTTON 118
DS_LAST_OP_INFO 124
DS_LOCATIONPOPUP 124
DS_NEWPASSWORD 117
DS_OBJECTCLASS 114
DS_OLDPASSWORD 118
DS_POSTEDVALUE 115
DS_SAVEBUTTON 117
DS_SEARCHDESC 115
DS_SORTENTRIES 115
DS_VIEW_SWITCHER 114
ENDHTML 123
entry-related 60, 110
GCONTEXT 109
HELPBUTTON 123
IF/ ELSE/ ELIF/ ENDIF 120, 125
in object class templates 60
INCLUDE 123
INCLUDECONFIG 124
miscellaneous 122
PCONTEXT 109
TITLE 123
used for changing colors on gateway forms 88
Directory Express
.conf file for 23
extended search result 24
pb.conf 51
search result 24
search support for user ID 73
Directory manager
authenticating as 39
configuring DN for 38
Directory Server
gateways installed with 22
updating gateway with updatedsgw script 27
updating the gateway with changes to 31
dirmgr parameter 95
DN formats
configuring 59
modifying default DN format 59
UID-based 59
dnedit CGI 128
dosearch CGI 129
dsgwfilter.conf 52, 68
dsgwsearchprefs.conf 51, 67

E

edit CGI 128
enable-aim-presence parameter 95
Entry types, see Object class templates

G

Gateway
and root suffix 33
configuring for a web server 33
creating a new instance 36
HTTP server requirements, see also HTTP Configuration 23
installed with Directory Server 22
running in high-usage networks 32
support for multiple instances 25
what is 21
Gateway CGIs 26
Gateway clients, see HTTP clients
Gateway cloning 30, 37
Gateway configuration
.conf file 37
configuring Communicator for preferred language 41
configuring gateway clients 40
configuring the browser for preferred language 41
default port setting 37
seting up directory manager 38

setting suffix for adding entries 39
setting up SSL support 40
Gateway configuraton
customizing Communicator's LDAP settings 42
Gateway files
.conf file 51
.conf file configuration 37
banner files 54
banner files, see also Banner files
dsgw.conf 51
dsgwsearchprefs.conf 51
files controlling gateway functionality 49
invoking CGIs in script files 127
object class templates 52
object class templates, see also Object class templates
pb.conf 51
script files 53
script files, see also Script files
search configuration files 51
search configuration files, see also Search configuration files
search result templates 53
search result templates, see also Search result templates
Gateway forms
appearance of 83
button images 84
changing color of table headings 89
changing table background color 90
color schemes 86
files controlling color 86
hexadecimal color values 88
on default gateway 25
Gateway object classes 60
Gateway object classes, see also Object class templates
Gateway performance on Red Hat Administration Server 32
Gateway planning, see Installation planning
Gateway search filters, see Search filters
Gateway search objects, see Search objects
Gateway search preference, see Search preferences
Gateway search results, see Search result templates
Gateway security
protecting bind DN and password 30
GET operations 26
gwnametrans parameter 97

H

htmldir parameter
Parameters
htmldir 97
HTTP clients
configuring Communicator for preferred language 41
configuring the browser for preferred language 41
customizing Communicator's LDAP settings 42
displaying non-English alphabets
language support for 41
Unicode and Latin-1 character sets 41
HTTP server
name trans mapping 33
HTTP server configuration
add additional CGI directory 34
add additional document directory 35
and gateway root suffix 33
change permissions of cookie directory 35
name translation mapping 33
HTTP server recommendations
for gateway 31
Red Hat Administration Server performance 31

I

ignoreAcceptpCharsetFrom parameter 97
include parameter 98
Installation planning 29

L

lang CGI 127
location parameter 56, 57, 98
location-suffix parameter 99

N

Name translation mapping 33
newentry CGI 129
newtype parameter 56, 100
NLS parameter 101
Non-anonymous searching 27

O

Object class templates
  adding attributes to 61
  adding search support for 74
  adding template for a child of a parent class 63
  and directives 60
  and entry types 55
  and entry-related directives 60
  attributes 59
  configuring DN formats for 59
  considerations for adding new 57
  creating a new parent object class 64
  creating new search object 75
  creating new search object, see also Search objects
  default object class templates 60
  default template files 52
  deleting attributes from 61
  extending 62
  mapping locations and entry types 57
  structure 108
orgchart-attrib-farleft-rdn parameter 101

P

Parameters
  authlifetime 91
  baseurl 92
  binddnfile 93
  changeHTML 93
  charset 93
  configdir 94
  dirmgr 95
  enable-aim-presence 95
  gwnametrans 97
  ignoreAcceptCharsetFrom 97
  location 56, 57, 98
  location-suffix 99
  newtype 56, 100
  NLS 101
  orgchart-attrib-farleft-rdn 101
  securitypath 102
  template 56, 102
  url-orgchart-base 102
  vcard-property 103
Port setting 37
POST operation
  using PCONTEXT 26

R

Root processes 31
Root suffix 33

S

Script files
  expressions for invoking CGIs in 127
  for Gateway forms 53
Search attributes 69
  adding 73
  adding search support 73
  search match types 73
  specifying 72
search CGI 127
Search configuration files
  dsgwfilter.conf 52, 68
  dsgwsearchprefs.conf 51, 67
Search filter
  for new object class 77
Search filters
  modifying 76
  user data patterns 77
Search objects
  creating 74

creating new 75
Search preferences
extending 74
Search result templates
customizing 78
default templates and related object classes 53
modifying 80
removing information from 81
Search results
display of 78
Search scope
base 69
defining 68
onelevel 69
subtree 69
Search support for user ID in Directory Express 73
securitypath parameter 102
serverID 18
serverRoot 18
SSL support 40
Standard search page 25

T

template parameter 56, 102

U

unauth CGI 128
updatedsgw script 27, 31
updating title.gif 84
url-orgchart-base parameter 102
UTF-8 43

V

vcard-property parameter 103

List of Tables

Table 2-1 Location of Gateway Files 30
Table 4-1 Gateway File Types and Locations 50
Table 4-2 Default Template Files and Related Object Classes 52
Table 4-3 Gateway Forms and Corresponding Script Files 53
Table 4-4 Default Search Result Templates 53
Table 4-5 Banner Files Controlling Appearance of Gateway Forms 54
Table 5-1 Default Object Classes 60
Table 6-1 Valid Search Scopes 69
Table 6-2 Bit Positions and Corresponding Search Match Types 73
Table 6-3 Default Search Results for Search Objects 79
Table 7-1 Button Images 85
Table 7-2 Files Controlling Appearance of Gateway Forms 86
Table 7-3 Colors and Related Attributes for BODY Tag 87
Table 7-4 Sixteen Standard Colors and Their Hexadecimal Values 89
Table B-1 DS_ATTRIBUTE: Display of syntax Argument 111
Table B-2 DS_ATTRIBUTE: Display of type Argument 111
Table B-3 DS_ATTRIBUTE: Display of options Argument 111
Table B-4 Conditions Supported for ELSE and ENDIF 121

Examples

HTML and configuration directories specified in pb.conf 6
Embedding GCONTEXT in a Link 7
Specifying PCONTEXT in an HTML Form 7
Setting up .conf file and directories for new gateway instance 21
Changing LDAP port in the baseurl parameter 23
Specifying the Path for Certificate Database 25
Specifying SSL Communication 25
Creating a locale for Chinese (zh) translation 33
Mapping of Locations and Entry Types in dsgw.conf 46
Adding Initials attribute to orgperson object class 49
Deleting mobile phone attribute from orgperson object class 50
Adding template for subclass siroePerson 51
Search Attributes for Person Entry 59
Extending Search Preferences to Include Pager Number 61
Editing dsgwsearchprefs.conf to Extend a Search Preference 62
dsgw-people entry configured to search for dateofBirth values 63
Creating a new dsgwsearchprefs.conf entry for OrgRole 63
Creating a new dsgwfilter.conf entry for orgrole 64
Search Filter for Email Address Searches 64
Modifying a search filter pattern 65
Adding OrgUnit to Search Results for People 67
Changing color settings using COLOR directives 74